Thomas Dachsel

ZENEKÖNYV

a Commodore 64-eshez

DATA BECKER – NOVOTRADE

Thomas Dachsel

ZENEKÖNYV
a Commodore 64-eshez

DATA BECKER – NOVOTRADE

A mű eredeti címe: Das Musikbuch zum Commodore 64 (1984)
Fordította: az APEX Szervező-Szolgáltató GMK munkacsoportja (Gyöngyös)
Lektorálta: Dr. Szidarovszky Ferenc
Felelős szerkesztő: Rochlitz András

A kiadásért felel: RÉNYI GÁBOR, a NOVOTRADE Rt. igazgatója
Kiadványmenedzser: BÉKÉS TAMÁS
Főszerkesztő: TARR KÁLMÁNNÉ
Műszaki szerkesztő: HAJDÚ ÁRPÁD

Szedte: Fényszedő Központ
Készült: Nyírségi Nyomda, Nyíregyháza
Budapest, 1986
ISBN 963 02 4429 2

FONTOS MEGJEGYZÉS

A könyvben szereplő kapcsolások, eljárások és programok közlései a bejelentett szabadalmak figyelembevétele nélkül kerülnek ismertetésre. Kizárólag amatőr és tanulási célokat szolgálnak, egyéb célokra történő felhasználásuk tilos!
A szerzők a közölt műszaki adatokat, kapcsolásokat és programokat a legnagyobb gondossággal állították össze. Ennek ellenére előfordulhatnak hibák, amelyekért a DATA BECKER GmbH sem garanciát, sem jogi felelősséget nem vállal.
A könyvvel kapcsolatos bármilyen észrevételt a szerzők köszönettel vesznek.

TARTALOMJEGYZÉK

1. BEVEZETÉS A SZÁMÍTÓGÉPES ZENE VILÁGÁBA

Kedves Olvasó!
Bizonyára azért vette meg ezt a könyvet, hogy megtudjon néhány dolgot a C–64 típusú személyi számítógép zenei lehetőségeiről.
Bizonyára érdekli Önt, hogyan lehet a számítógépen dallamot játszani, valamint hogyan lehet olyan hangeffektusokat létrehozni, melyeket egy játékprogramnál is fel lehet használni. A számítógéppel mindezeken felül komponálhat is, továbbá olyan hangzásokat is létrehozhat, melyekre ebben az árkategóriába tartozó számítógép egyike sem képes.
Akad olyan ember, aki azonnal azt mondaná: „Ez mégiscsak számítógépzene, ami nem lehet több, mint játék." Honnan származik ez a téves felfogás a „számítógépből jövő" zenével szemben? Mi is az, ami olyan különleges ebben a zenében? Mielőtt választ adhatnánk ezekre a kérdésekre, át kell gondolnunk, hogy milyen zenét tartunk „normálisnak".
A vélemények természetesen különbözőek lesznek. Mindenesetre létezik olyan „normális" zene, amely közelebb áll hozzánk, mint amit a számítógép játszik.
A kérdésekre általánosan válaszolva: a zene olyan valami, melyet ember készít és hallható eredményt szolgáltat számunkra.
Ezen definíció szerint a légkalapács is zenét szolgáltat, vagyis a definíciót le kell határolni.
Próbáljuk meg: tehát csak az a zene, ami dallamosan cseng a fülünk számára. Ezek után azonban az Európán kívüli zenék nagy részét, pl.: az indiait, mely számunkra furcsán hangzik, valójában nem nevezhetnénk zenének. Lehetne hangerőkorlátokat is meghatározni pl.: ami 80 decibelnél hangosabb, az már nem zene. De mi a helyzet a Hard-Rock koncertek esetében, ahol a 80 decibel teljesen normális? Az idősebb emberek gyakran hangoztatják is véleményüket, miszerint „ez már nem is zene". A fiatalok számára az ilyen hangerő még kellemes hangzást biztosít. Összefoglalva: Olyan, hogy „normális" zene nem létezik, általános érvényű meghatározásként semmiképp. Bár mindenkinek van elképzelése a „normális" zenéről, ez viszont függ a kortól, a kultúrkörnyezettől amelyben élünk, és nem utolsósorban a személyes zenei ízlésünktől, amelyről nem kívánunk vitatkozni.

Mindezek után visszatérhetünk a definíciónkhoz, hogy meghatározzuk a zene általános tulajdonságait és megállapítsuk, hogy miért is „különleges" a számítógépes zene. Minden zenedarab valamilyen ötlet szülötte, melyet nem láthatunk, hallhatunk vagy érzékelhetünk. Az ötletet keresi mindenki, aki meg akar érteni egy komponált darabot.

Ha a zeneszerzőnek nem jut eszébe semmi, akkor nem tud zenét szerezni. Ahhoz viszont, hogy egy zeneszámot elő lehessen adni, meg kell komponálni.

Tegyük fel, hogy a zeneszerző ötletet keres, várja az ihletet. Hosszú időn keresztül ül a zongora mellett, és ismert dallamokat vagy skálákat játszik. Azonban minél jobban igyekszik, hogy valami újat alkosson, annál inkább nem jut eszébe semmi. A meghiúsult kísérlet után nem játszik, nem gondolkodik tovább, hanem valami mást csinál: elmegy sétálni vagy megnéz egy kiállítást (az ilyesmit inspirációnak nevezzük). Előfordulhat, hogy időközben eszébe jut egy szimfónia teljes tétele. Ilyenkor ismét leül a zongora mellé és lejegyzi a tételt.

A zenedarab leírása feltétlenül fontos és szükséges tevékenység. Egyetlen szerző sem képes anélkül komponálni, hogy legalább jegyzeteket ne készítsen.

A zene feljegyzésének módja kultúrkörönként és hangszerenként más és más. Minden hangszernek más a hangterjedelme, azaz csak pontosan meghatározott számú hangot képes megszólaltatni. Egy nagybőgő csak mély hangokat, egy hegedű csak magas hangokat tud játszani. A zongora esetében a teljes skála rendelkezésre áll.

Tehát minden hangszernek megvannak a sajátosságai, ami alapján a hangjegyeket lejegyzik. Nagybőgő esetén a hangokat basszuskulcsban, a hegedű esetében violinkulcsban írjuk le. A zongora esetében mindkét kulcsot használjuk.

A hangjegy, a violinkulcs, a basszuskulcs stb. a hangjegyírás fogalmai, Európában időszámításunk kezdete után kb. 1000 évvel alakultak ki kezdetleges formában és a mai napig fejlődtek tovább. Ez azonban nem azt jelenti, hogy csak egyetlen hangjegyírás létezik. Más kultúrákban hasonló komplex rendszerek alakultak ki a zenei információ írásos rögzítésére, hasonlóan a betűk, szavak, mondatok írására. Itt a zene és a beszéd érdekes párhuzamával találjuk szemben magunkat: mindkét emberi vívmány egy írásrendszert igényel, hogy meghatározott időtartamon keresztül rögzíthető legyen.

A kottaírás nem képes minden lényeges és használandó információt egzakt módon leírni a zenedarab jellemzéseként, ezért a lejátszásnak különféle módjai léteznek. Ezt a darab interpretációjának (értelmezésének, előadásának) nevezzük. Az interpretációra különösen a klasszikus zene esetében kell ügyelni, mert a lényeg az, hogy egy darabot úgy játsszunk el, ahogy azt a zeneszerző elképzelte. Megkíséreljük, hogy a játszott darab „hiteles" maradjon, ami a klasszikus zene területén szükségszerű.
A klasszikus zenekari művek esetében a karmester az összekötő kapocs a zeneszerző és a hallható zenemű között. Neki kell megkísérelnie a sok pontatlanság és a homályosság kézben tartását, melyek a kottaírással történő lejegyzés során csak hiányosan leírt zeneművet eredményezhetnek.

A jazzben nem kell törekedni a hitelességre, mert az ismert műveket gyakran átírják. A jazzben az átdolgozásra jut hasonló szerep, mint a klasszikusok esetében a karmesterre. Az átdolgozó ugyanis az ismert zenemű átvételén túl azt egyúttal saját elképzelése szerint megváltoztatja, új részleteket fűz hozzá és módosít a hangszerelésen is. Az átdolgozó, ellentétben a karmesterrel, együtt játszik a zenekarával, ráadásul egy markáns szólóhangszeren.

A pop- és a rockzene esetében nincs szükség karmesterre, mert az együttesek többsége közösen komponált saját zeneművet ad elő, vagyis mindenki a saját hangszeréért felelős. Ezáltal minden zenész egyben önmaga karmestere vagy átdolgozója is. Ahhoz, hogy a hangszerek illeszkedjenek egymáshoz, a próbák során a zenészeknek kell egymás között megegyezniük.

Az interpretáló vagy előadó a másik összekötő kapocs a kész zenemű és a zeneszerző között, akit első látásra „zenésznek" neveznénk. A valóságban az interpretáló utánozza a mások által már előzőleg kitalált történéseket, tehát reá csak reprodukáló funkció hárul. Ez alól kivétel az improvizáció, ahol az előadó vagy a zenész aki improvizál, egy személyben zeneszerző és átdolgozó is.

Azon zenészeknek, akiknek improvizálniuk kell, a hangszereiket mesteri módon kell ismerniük és kezelniük, mert improvizációjukat technikai nehézségek látszata nélkül azonnal hallható eredménnyé kell átalakítani. Több zenész együtt csak akkor képes improvizálni, ha nagyon jól ismerik egymás zenei képességeit és a hirtelen támadt zenei ötletekre azonnal képesek reagálni.

A zeneszerző és a kész zenemű között az utolsó, de legfontosabb összekötő kapocs a hangszer. Nem elég, ha a zenész a hangszerén tud játszani, még ki kell fejlesztenie azt a technikai tudást, amellyel a kottában szereplő előjegyzéseknek eleget tud tenni. Ezt a képességet a zenészek sokéves, gyakran rendkívül fáradságos tanulással szerzik meg.
Ezért ritkán fogunk olyan személyt találni, aki egy személyben jó komponista és hangszerén is tökéletesen tud játszani. Egy zeneszerző mindenképpen jó átdolgozó is, azonban a technikai követelményekhez mindig zenészre van szükség, aki a kompozíciót elő tudja adni. Különösen így van ez a klasszikus zeneszerzők esetében, hisz nincs olyan zeneszerző, aki a zenekar valamennyi hangszerén játszani tudna. Az úgynevezett multiinstrumentalistából, aki egynél több hangszeren is képes játszani, kevesebb van, mint aki (jobban vagy kevésbé jól) komponálni tud.

Miután nagy vonalakban áttekintettük a zeneszerző, az átdolgozó, az előadó, a hangszer és a zenemű viszonyát, nézzük meg, mi is történik másképpen a számítógépes zene esetében.

Mindkét zenében a komponista az ember marad. Nincs az a számítógép, amelyik meg tudna szólalni, ha kitennénk egy sivatagba, és felszólítanánk, hogy játsszon el egy zeneművet. Továbbá a zeneszerzés folyamata sem a számítógépben zajlik. A számítógépes programozók rendszerint egy ismert zeneművet választanak, melyet már nem kell megkomponálni azért, hogy átültessék egy számítógépre.

Itt kell megemlíteni a számítógépes zene egy sajátosságát, miszerint vannak olyan programok, melyekkel számítógépes zenét lehet „komponálni". Ezek a programok „véletlen" dallamokat képesek „komponálni" meghatározott adatok alapján.

Ennek ellenére a számítógép ebben az esetben sem végez kreatív tevékenységet, itt is az ember végzi a kreatív munkát azzal, hogy egy úgynevezett algoritmust hoz létre, ami szimulálni képes a zene komponálását vagy improvizálását, ha az algoritmust a számítógép nyelvére lefordítjuk.

Tehát egy zenei programra van szükség ahhoz, hogy a számítógép képes legyen lejátszani egy zeneművet. Ezt a programot a programozó által létrehozott hosszú táblázat jellemzi, melyben ugyanúgy, mint a kottában, minden információ megtalálható a hangok magasságára és időtartamára vonatkozóan. A számítógép azonban nem rendelkezik 88 különböző hanggal, mint a zongora és egyidejűleg csak meghatározott számú hang lejátszására alkalmas. Ezért a programozónak, mivel időeltolással ő a karmester is, át kell gondolnia és a számítógép által érthető formában, valamint időrendben kell összeállítania az összes zenei információt.

A programozó tehát azonosítható egy átdolgozóval vagy karmesterrel, a zenei program pedig a zenemű interpretálójaként fogható fel. Ugyanis ez a zenei program „tud" mindent, ami egy zenemű lejátszásához szükséges.

De mi is a hangszer a számítógép esetében? Hiba lenne a számítógépet azonosítani vele, mert a számítógépnek csak meghatározott alkatrészei azok, melyek a hallható hangot előállítják.
A Commodore 64-es esetében ezen alkatrészek a 6581-es hang chip és a 6510-es mikroprocesszor. A chipek és más alkatrészek közötti vezetékek (buszok) nélkül azonban ez a számítógép sem lenne képes hang létrehozására.

Álljunk csak meg egy pillanatra:
A Commodore 64-es, mint számítógép nem is hoz létre semmiféle hangot! Bármilyen zenei programot futtathatunk , ha a gép nincs hozzákapcsolva egy televízióhoz vagy egy sztereó berendezéshez*, nem fogunk semmit hallani. Ez viszont a hangszer fogalomkörére is vonatkozik, úgyhogy a számítógépet, a hangszórót, a televíziót és az összekötő kábelt is ennek a hangszernek a részeként kell kezelni.

*sztereó berendezésen hifi-tornyot, de legalább egy hangszórópárral ellátott erősítőt értünk. (ford. megj.)

Mindezeken keresztül eljutottunk ahhoz.a furcsasághoz, ami miatt a számítógépes zene „idegenül" hat, vagyis nem látjuk a zenészt, aki a hangszeren játszik, és a hangszer, ami előttünk áll nem egészen olyan zeneszerszám, amilyet megszoktunk. A zenész különös módon olyan „valami", amit nem láthatunk: ez a zenei program, a „szoftver". A hangszer pedig a hang chip és mindazon eszköz, ami a hallható hang előállításához hozzátartozik, tehát a „hardver".

Továbbá a hangzás szempontjából is furcsán hat számunkra a számítógépes zene. Nemcsak a hangzások „mesterségesek", hanem az a mód is, ahogyan a gép játszik. Egyetlen zenész sem lenne képes pl. egy Bach fugát olyan biztosan és hosszantartóan játszani, mint a számítógép.
Egy zenei program naphosszat futtatható anélkül, hogy elfáradna a gép vagy egyszer is hibázna. Ellenkezőleg, mindig ugyanúgy nullaszekundum pontossággal játssza a hangjegyeket.

Ugyanakkor ebben rejlik a számítógépes zene veszélye is, mert a programozókat könnyű rávenni arra, hogy csak az egzakt hangjegyeket használják a programozás során. Ezzel magyarázható, hogy Bachnak miért fordítják le olyan sok művét zenei programmá. A Bach zene ezáltal egy bizonyos bájt nyer, mivel maga is egzakt módon és matematikai szempontok alapján komponálódott. Ez a zene viszont sterilen és személytelenül hangzik.

Természetesen és kifejezően hangzó zenei programok írásához egyszerre kell jó programozónak és jó zenésznek lenni.

Tehát, hogy a számítógép számunkra megszokott zenét szolgáltasson, a legnehezebb dolgok egyike. De nem ez a számítógépes zene feladata. Bizonyos területeken azonban előnyünkre válnak a számítógépes zene szokatlan tulajdonságai, és itt már egy új zenei földre léptünk.

A számítógépes technológia használatával teljesen új hangszereket lehet kifejleszteni, melyek olyan lehetőségeket biztosítanak számunkra, amiket egyetlen hagyományos hangszer sem kínálhat. Ezek többek között a szintetizátorok és a szekvenciátorok. Az új digitális technológia elvének alkalmazásával bővült ezen hangszerek köre is.

A Commodore 64-es szintetizátor chipjének kifejlesztése is ezen technológia elvén történt. Az ehhez a chiphez hasonló „zeneszerszámok" az elektronikus zene alapját képezik. Olyan előadók és együttesek mint pl. Jean-Michel Jarre, Klaus Schulze, a Kraftwerk és a Tangerine Dream ezen technológia alkalmazása nélkül ma már nem lennének képesek új zene komponálására. A korábbi digitális technikához képest elképzelhetetlen a hangzás sokrétűségének bővülése. A hagyományos hangszereket hangzás szempontjából lehet utánozni, ezen túl a digitális hangszerek erőssége a variációs lehetőségükben rejlik.

A Commodore 64-es zenei képességei a piacon mércét szabnak a többi hasonló készülék számára is. Mintegy 10 évvel ezelőtt jelent meg a Moog cég népszerű kisszintetizátora, a „minimoog". Ez sok hasonló tulajdonsággal rendelkezett már, mint a Commodore 64-es hang chipje. Megjelenésekor a Moog szintetizátor sok ezer márkába került annak ellenére, hogy kevesebbet tudott, mint a 6581-es hang chip. Ma már egy bélyeg nagyságú chip is a Moog szintetizátor szolgáltatásainak többszörösére képes, mert rendelkezésre állnak az előbbi lehetőségek:

- három digitális oszcillátor
- nyolc oktáv oszcillátoronként
- oszcillátoronként négy keverhető hullámforma:
  fűrészfog-, háromszög-, négyszögimpulzus és fehérzaj
- az impulzusszélesség minden oszcillátornál külön-külön beállítható
- három burkológörbe generátor
- a hanglefutás minden fázisa külön állítható
- egy programozható szűrő (felül-, aluláteresztő és sávszűrő keverhető), mely minden oszcilátorra külön rákapcsolható, határ- és rezonanciafrekvenciával beállítható
- speciális effektusokhoz gyűrűs vagy ringmodulátor és szinkronizálás.

Próbáljuk meg ezeket a lehetőségeket megismerni, és a Commodore 64-es szintetizátorát megszólaltatni.

2. ELSŐ LÉPÉSEK A COMMODORE 64-ESSEL

A Commodore 64-esben lévő hang chipet próbáljuk meg játékra bírni. Mind a gép bekapcsolása után, mind néhány billentyű leütése után a hang chip néma marad. A kulcs a hang chiphez a POKE utasítás. Ezért nem kell csodálkozni azon, hogy a következő néhány sor nagyrészt csupa POKE utasításból áll:

```
0 REM **** P1. ****
1 :
2 :
3 A=54272:POKEA+24,15:POKEA+6,240:POKEA+4,33:P=0
4 POKEA+1,PAND255:P=P+1:IFPEEK(198)=0THEN4
5 POKEA+1,0:POKE198,0
READY.
```

A programot beírás után RUN-nal el lehet indítani. Most lehet hallani, hogy a Commodore 64-es milyen hangmagasságokra képes. A skála az egészen mély hangoktól az egészen magas hangokig terjed. Egy tetszőleges billentyű lenyomásával a program megáll. Ha a POKE A + 1,P AND 255 utasítást a POKE A + 1, 255 – (P AND 255)-re módosítjuk, akkor a hangskálát fordított irányban hallhatjuk.

Az alábbi példaprogramban mindkettő egyszerre hallható. Egy hang a legmélyebbről indulva a legmagasabbig folyamatosan változik, a másik pedig ellenkezőleg.

```
0 REM **** P2. ****
1 :
2 :
3 A=54272:POKE A+24,155:POKE A+6,240:POKE A+4,33:P=0
4 POKE A+6,240:POKE A+4,33+1:IF PEEK(198)=0 THEN 3
5 POKE A+13,240:POKE A+11,33
6 P=0
7 POKE A+1,P AND 255
8 POKE A+8,255-(P AND 255)
9 P=P+1:IF PEEK(198)=0 THEN 7
10 POKE A+1,0:POKE A+8,0:POKE 198,0
READY.
```

A PEEK (198) itt is a megnyomott billentyűk számát adja. A program tetszőleges billentyű lenyomása után megáll.

Tapasztalhattuk, hogy a hang chip milyen széles frekvencitartománnyal rendelkezik, amiből tetszés szerinti frekvenciákat lehet kiválasztani. Az alábbi példaprog-

ramban az utoljára lenyomott billentyű kódja határozza meg a frekvenciát. Mihelyt a SPACE (szóköz) billentyűt lenyomjuk, a program megáll.

```
0 REM **** P3. ****
1 :
2 :
3 A=54272:POKE A+24,15
4 POKE A+6,240:POKE A+4,65
5 POKE A+1,PEEK(197):POKE A+3,PEEK(162)/32
6 POKE A+2,(PEEK(162) AND 7)*32
7 IF PEEK(197)<>60 THEN 5
8 POKE A+1,0:POKE 198,0
READY.
```

A hang chip a zenei hangokon kívül zajokat is előállít. A következő példaprogramban a Commodore 64-es megszakításóráját ("JIFFY CLOCK") használjuk a hangok és a zajok frekvenciájának meghatározásához. A PEEK (162) ezen óra egyik időadata.

```
0 REM **** P4. ****
1 :
2 :
3 A=54272:POKE A+24,15
4 POKE A+6,240:POKE A+4,129
5 POKE A+1,31-(PEEK(162) AND 31):IF PEEK(198)=0 THEN 5
6 POKE A+1,0:POKE 198,0
READY.
```

A programban a zajok frekvenciája a fehérebb zajok felől a sötétebb zajok irányába változik. Ha az 5. számú sorból elhagyjuk a 31-et, akkor megfordul az egész.

A hangok és zajok módosítására a hang chip esetében egy másik lehetőség is rendelkezésre áll, mégpedig a *szűrő*. A szűrő esetében is meg lehet határozni egy frekvenciát, ami megadja, hogy a zaj milyen világosan vagy sötéten csengjen. Az alábbi példaprogramban a szűrő frekvenciáját először 0-ról a maximális értékig változtatjuk, majd a maximumról 0-ra csökkentjük. Az eredmény egy előretörő majd fokozatosan elcsendesedő zaj, ami a mozdony zúgására emlékeztet.

```
0 REM **** P5. ****
1 :
2 :
3 A=54272:POKE A+24,31
4 POKE A+6,240:POKE A+4,129
5 POKE A+1,200:POKE A+23,1
6 P=(PEEK(162) AND 31)*16:IF P>255 THEN P=(P-(P-255)*2)
7 POKE A+22,P
8 IF PEEK(198)=0 THEN 6
9 POKE A+1,0:POKE 198,0
READY.
```

Eddig két módszert mutattunk a hangfrekvencia meghatározására, egyik a számláló, a másik a billentyű megnyomása. Egy zeneműnek a Commodore 64-esen

történő lejátszásához meg kell határozni, hogy milyen frekvenciájú hangokat milyen hosszan kell játszani. Az alábbi példaprogram – mely megmutatja, hogyan is kell ezt csinálni – az Abba "Super Trooper" c. számának refrénjét játssza le.

```
0 REM **** P6. ****
1 :
2 :
3 A=54272:POK! A+24,15
4 POKE A+6,24∂:POKE A+4,33
5 READ L,H,D:IF L<0 THEN POKE A+24,0:END
6 POKE A,L:POKE A+1,H:FOR I=0 TO D:NEXT:GOTO 5
7 DATA 30,25,200,135,33,200,62,42,200,60,50,300,0,0,100
8 DATA 60,50,300,0,0,100,193,44,300,0,0,100,193,44,300
9 DATA 0,0,100,62,42,200,162,37,200,62,42,200,193,44,200
10 DATA 62,42,300,0,0,100,162,37,300,0,0,100,193,44,300
11 DATA 0,0,100,193,44,300,0,0,100,62,42,300,0,0,100
12 DATA 62,42,300,0,0,100,162,37,1100,0,0,100,193,44,300
13 DATA 0,0,100,193,44,300,0,0,100,62,42,300,0,0,100
14 DATA 62,42,300,0,0,100,162,37,1100,0,0,100,162,37,200
15 DATA 135,33,200,162,37,200,62,42,200,162,37,300,0,0,100
16 DATA 135,33,300,0,0,100,135,33,800,-1,0,0
READY.
```

Ez a rövid BASIC program megmutatja minden zeneprogram alapvető szerkezetét, mely a következő: egy rövid programrészlet (0–6-os sor) és egy igen hosszú adatsor (DATA sorok 7-től 16-ig).

A következő fejezet azzal foglalkozik, hogy az egy-, két- és háromszólamú zenemű a BASIC-ben hogyan programozható.

```
0 REM **** P7. ****
1 :
2 :
3 REM *** "LET IT BE"
4 REM
5 REM *** A TAROLOK NULLAZASA
6 REM
7 FOR I=54272 TO 54296
8 POKE I,0
9 NEXT I
10 REM
11 REM *** AZ ALKALMAZOTT JELOLESEK
12 REM
13 ELSO    =54272
14 HANGERO=ELSO+24
15 BE      =ELSO+5
16 KI      =ELSO+6
17 H1      =ELSO
18 H2      =ELSO+1
19 NYOMAS =ELSO+4
20 REM
21 REM *** A KEZDOERTEKEK BEALLITASA
22 REM
23 POKE HANGERO,15
24 POKE BE      ,23
```

```
25 POKE KI     ,123
27 REM
28 REM *** EGY HANG BEOLVASASA
29 REM
30 READ MAGASSAG,TARTAM
40 IF MAGASSAG=0 THEN END
47 REM
48 REM *** A FREKVENCIA KISZAMITASA
49 REM
50 F2=MAGASSAG/256:F1=MAGASSAG-F2*256
60 POKE H2,F2 :POKE H1,F1
67 REM
68 REM *** "BILLENTYU-LENYOMAS"
69 REM
70 POKE NYOMAS,33:FOR I=0 TO TARTAM*100
75 NEXT
77 REM
78 REM *** "BILLENTYU-ELENGEDES"
79 REM
80 POKE NYOMAS,32:FOR I=0 TO TARTAM*100
85 NEXT
87 REM
88 REM *** A KOVETKEZO HANG LEJATSZASA
89 REM
30 GOTO 30
97 REM
38 REM *** AZ OSSZES HANGMAGASSAG
99 REM     ES IDOTARTAM LISTAJA
100 DATA 6430,1
110 DATA 6430,1
120 DATA 6430,3
130 DATA 6430,1
140 DATA 7217,2
150 DATA 5407,2
160 DATA 6430,1
170 DATA 6430,2
180 DATA 8583,3
190 DATA 9634,2
200 DATA 9634,1
210 DATA10814,2
220 DATA10814,3
230 DATA 9634,2
240 DATA 9634,2
250 DATA 8583,1
260 DATA 8583,5
270 DATA10814,1
280 DATA10814,2
290 DATA11457,3
300 DATA10820,2
310 DATA10814,2
320 DATA 9634,4
330 DATA10814,1
340 DATA 9634,1
350 DATA 9634,1
360 DATA 8583,11
370 DATA10814,1
380 DATA 9634,2
390 DATA 8534,5
400 DATA10814,1
410 DATA12860,2
420 DATA14435,5
430 DATA12860,1
440 DATA12860,2
450 DATA10814,3
460 DATA 6439,2
470 DATA 6439,1
480 DATA 6439,2
490 DATA10814,9
500 DATA10814,1
510 DATA10814,2
520 DATA11457,3
530 DATA10820,2
540 DATA10814,2
550 DATA 9634,4
560 DATA10814,1
570 DATA 9634,1
580 DATA 9634,1
590 DATA 8583,15
999 DATA     0,0
READY.
```

3. ZENEPROGRAMOZÁS COMMODORE BASIC-BEN

3.1. „Let it be" – Egy egyszólamú zeneprogram

Bebillentyűztük a "Let it be" programot?
Ha ismerjük a BASIC programozást, nem lesz vele problémánk. Valószínűleg azt is tudjuk, hogy a REM utasítást tartalmazó sorokat el lehet hagyni. Ezzel ugyan csökken a beviteli munka, de egyúttal a program is áttekinthetetlenebb lesz. Minden BASIC parancsot és programsort a RETURN billentyűvel kell zárni, annak egyszeri megnyomásával. A bevitt programot a RUN paranccsal lehet indítani.

Nézzük meg, hogy a program milyen típusú BASIC utasításokat tartalmaz. Található benne FOR NEXT ciklus, rengeteg POKE utasítás és a 100-as sortól még több DATA utasítás. A BASIC nyelvű zeneprogramozáskor ezekkel az utasításokkal újra és újra találkozni fogunk. A továbbiakban megkíséreljük bemutatni, elmagyarázni ezen utasítások működését.

A 6581-es hang chip az alapvető építőelem a C–64-esben, mely hangot állít elő. A működés elvének könnyebb megértése végett a hang chipet egy irodaháznak képzeljük. Az irodaház 29 szobával és minden szobában meghatározott számú alkalmazottal rendelkezik. A C–64-es bekapcsolása után az alkalmazottak azt az utasítást kapják, hogy ne csináljanak semmit, amíg újabb utasítást nem kapnak. A következő utasítás a POKE szóval kezdődik, majd egy ötjegyű szobaszámmal folytatódik. Ez a szobaszám az 54272 és 54300 közötti lehet.

A szobaszámot követő vessző azt jelöli, hogy az utána következő utasítás kerül a vessző előtt megadott szobába. Ez az utasítás egy 0 és 255 közé eső számból áll. Ennek oka az, hogy a számítógép csak számokat tud kezelni.

Az előbbiek során rögzített számítógépes program is, mely a memóriában található, 0 és 255 közé eső számok sorozata. A "Let it be" program 7–9-es soraiban a szobaszám 0-s utasítást kapott, ami valóban a 0 és a 255 közé esik. A nulla jelentése, hogy az alkalmazottak hagyják abba az eddig végzett tevékenységüket, és csak a következő utasításra figyeljenek.

A számok megjegyzése az ember számára nehézkes, ezért a BASIC-ben lehetőség van arra, hogy a számok helyett betűket használjunk, amit *változó*nak nevezünk. A számítógépnek érthető nyelven meg kell mondani, hogy "Hallgass rám! Az 54272-es szám helyett a továbbiakban az "ELSŐ" szót használjuk." De nem ilyen

formában, mert a válasz "Syntax Error" lenne, ugyanis a számítógép nem értené meg a kívánságunkat. A bevitel ELSŐ = 54272 formája azonban már érthető. Ettől fogva a gép tudja, hogy ha ELSŐ kerül bevitelre, az valójában 54272-t jelent. Ezt mindaddig így kezeli, amíg a számítógépet ki nem kapcsoljuk, vagy olyan "súlyos" utasítást nem adunk, mint a CLR vagy a NEW. A számítógép akkor is elfelejti ezt az utasításunkat, ha helyette új programsort viszünk be, vagy a régit módosítjuk. Ez az utasítás a "Let it be" program 13-as sorában található. Az összes további BASIC mintaprogramban használni fogjuk az ELSŐ = 54272 utasítást.

Az ELSŐ tehát a "hang chip iroda" első szobája. A program 14–19-es sorai a többi szobát jelölik. A 23–25-ös sorokban az előbbi szobákban lévő alkalmazottak utasítást kapnak, melyek a program során már nem változnak. A program további része néhány olyan ciklusból áll, melyek a program végén található listából egy-egy hangjegyet kiolvasnak és egy ideig várakoznak.

Mielőtt a további részleteket taglalnánk, tisztában kell lennünk a zeneprogram szerkezetével, felépítésével. Ez a struktúra minden zeneprogramban megtalálható, függetlenül attól, hogy melyik könyvben szerepel.

A hang chipet sok szobából álló irodaháznak tekintettük. A zeneprogram viszont egy irodai kézbesítő, "amely" szobáról szobára jár és minden alkalmazottnak az új rendelkezésről ad egy papírlapot. A programozó az iroda vezetője, és ennek megfelelően adja a kézbesítőnek a rendelkezések listáit. Ez az a szöveg, melyet programként bebillentyűzünk a gépbe.

A LIST parancs segítségével bármikor kiírathatjuk ezt a programlistát. A lista elkészülte után lehet indítani az irodai kézbesítőt a RUN paranccsal. A RUN/STOP billentyű segítségével bármikor leállíthatjuk, a CONT utasítással pedig továbbküldhetjük a "küldöncöt".

De hogyan is néz ki egy ilyen lista? Nézzük meg példaként a „Let it be" program listáját. A lista vizsgálata során azonnal szemünkbe tűnik a program kettéválása az adatokra, és a program folyamatára. Ez a legtöbb program tulajdonsága, de a zenei programoknál ismerhető fel a leginkább.

A példaprogram esetében a 0-tól a 90-es sorig terjed a tényleges programrész, a 100-as sortól kezdve pedig az adatrész található. Az egyszerű zeneprogramoknál a két rész közel egyforma hosszúságú. Bonyolultabb zeneprogramok esetében, melyekkel a későbbiek folyamán még találkozunk, az adatrész kb. négyszer hosszabb. Valójában mi is az oka ennek?

Egy zeneprogram nagyobb részben a program által lejátszandó zenemű zenei információiból áll. Mivel a program mindig csak egy hangot játszik, ezért viszonylag rövid. A zenei információ tárolása adatblokkban történik.

Az adatblokk megadására a DATA utasítás szolgál, ami biztosítja, hogy a programban bárhol elhelyezhetjük az információinkat. Sok esetben a nagyszámú DATA utasítás tömbben helyezkedik el. A mintaprogramban minden hangjegyet külön sor tartalmaz. Lehet azonban egy sorba több hangjegyet is írni, ami viszont kevésbé áttekinthető, pl.:

100 DATA 6430,1, 6430,1, 6430,3 ... stb.

Valójában milyenek is ezek az adatok?
Megfigyelhető, hogy minden esetben két számot találunk párban. Az első számjegy négy- vagy ötjegyű lehet, és a hang magasságát (frekvenciáját) jelenti. A használatos hangfrekvenciák a 4. fejezetben levő táblázatban találhatók.

A második szám a hang időtartama, ami 1 és 15 közé eső érték lehet. Az 1 a nyolcadot jelenti, ami rövid időnek tekinthető. Az összes többi hangjegyérték egyszerű összeadással képezhető: a 2 a negyed-, a 4 a félhangot jelenti stb.

Az előbbi mintaprogramban a DATA utasítások értékeinek módosításával teljes egészében új zeneszám, a "Hey Jude" dallama adódik.
Ennek listája a következő:

```
0 REM **** P8. ****
1 :
2 :
3 REM *** "HEY JUDE"
4 REM
5 REM *** A TAROLOK NULLAZASA
6 REM
7 FOR I=54272 TO 54296
8 POKE I,0
9 NEXT I
10 REM
11 REM *** AZ ALKALMAZOTT JELOLESEK
12 REM
13 ELSO    =54272
14 HANGERO=ELSO+24
15 BE      =ELSO+5
16 KI      =ELSO+6
17 H1      =ELSO
18 H2      =ELSO+1
19 NYOMAS =ELSO+4
20 REM
21 REM *** A KEZDOERTEKEK BEALLITASA
22 REM
23 POKE HANGERO,15
24 POKE BE      ,122
25 POKE KI      ,120
27 REM
28 REM *** EGY HANG BEOLVASASA
29 REM
30 READ MAGASSAG,TARTAM
40 IF MAGASSAG=0 THEN END
47 REM
48 REM *** A FREKVENCIA KISZAMITASA
```

```
49 REM
50 F2=MAGASSAG/256:F1=MAGASSAG-F2*256
60 POKE H2,F2 :POKE H1,F1
67 REM
68 REM *** "BILLENTYU-LENYOMAS"
69 REM
70 POKE NYOMAS,17:FOR I=0 TO TARTAM*50
75 NEXT
77 REM
78 REM *** "BILLENTYU-ELENGEDES"
79 REM
80 POKE NYOMAS,16:FOR I=0 TO TARTAM*50
85 NEXT
87 REM
88 REM *** A KOVETKEZO HANG LEJATSZASA
89 REM
90 GOTO 30
97 REM
98 REM *** AZ OSSZES HANGMAGASSAG
99 REM     ES IDOTARTAM LISTAJA
100 DATA 8583,4
110 DATA 7217,10
120 DATA 7217,2
130 DATA 8583,2
140 DATA 9634,2
150 DATA 6430,12
160 DATA 6430,2
170 DATA 7217,2
180 DATA 7647,4
190 DATA11457,6
200 DATA11457,2
210 DATA10814,2
220 DATA 8583,2
230 DATA 9634,2
240 DATA 8583,1
250 DATA 7647,1
260 DATA 7217,10
270 DATA 8583,2
280 DATA 9634,2
290 DATA 9634,4
300 DATA 9634,2
310 DATA12860,1
320 DATA11457,2
330 DATA10814,2
340 DATA11457,1
350 DATA 9634,2
360 DATA 8583,8
370 DATA 5728,2
380 DATA 6430,2
390 DATA 7217,2
400 DATA 9634,4
410 DATA 8583,4
420 DATA 8583,2
430 DATA 7647,2
440 DATA 7217,2
450 DATA 5407,2
460 DATA 5728,4
```

```
470 DATA 5728,17
480 DATA 5728,1
490 DATA11457,1
500 DATA10207,2
510 DATA 9634,3
520 DATA 8583,2
530 DATA 8583,2
540 DATA 7647,1
550 DATA 9634,5
560 DATA11457,2
570 DATA 9634,6
580 DATA11457,2
590 DATA 7647,6
600 DATA11457,2
610 DATA 9634,4
620 DATA 8583,2
630 DATA 7647,2
640 DATA 8583,5
650 DATA 9634,3
660 DATA 8583,4
670 DATA 7647,4
680 DATA 7217,2
690 DATA 6430,2
700 DATA 5728,17
710 DATA 5728,1
720 DATA 7647,1
730 DATA 9634,2
740 DATA11457,3
750 DATA 9634,2
760 DATA 9634,2
770 DATA 8583,1
780 DATA 9634,5
790 DATA11457,2
800 DATA 9634,6
810 DATA11457,2
820 DATA 7647,6
830 DATA11457,2
840 DATA 9634,4
850 DATA 8583,2
860 DATA 7647,2
870 DATA 8583,6
880 DATA 9634,2
890 DATA 8583,2
900 DATA 7647,5
910 DATA 7217,4
920 DATA 6430,1
930 DATA 5728,6
940 DATA 5728,2
950 DATA 8583,2
960 DATA 9634,2
970 DATA10207,2
980 DATA 9634,2
990 DATA10207,4
1000 DATA10814,2
1010 DATA11457,2
1020 DATA12860,4
1030 DATA12860,8
1040 DATA    0,0
READY.
```

3.2. „Hey Jude” – A BASIC zeneprogram szerkesztése

A „Hey Jude" program bevitele abban az esetben, ha a „Let it be” program a memóriában még megtalálható, egyszerű lesz. A program szerkezete azonos, csak az adatokban van változás.

Lépésről lépésre nézzük meg a különbségeket.
A 0–90-es sorok mindkét esetben (néhány szám értékének eltérésével) gyakorlatilag azonosak.

- Vigyük be a LIST 24-et,
- Módosítsuk a 23-as számot 122-re, és
- Nyomjuk meg a RETURN billentyűt.

- Vigyük be a LIST 25-öt,
- Módosítsuk a 123-as számot 120-ra, és
- Nyomjuk meg a RETURN billentyűt.

– Vigyük be a LIST 70-et,
– Módosítsuk a 33-as számot 17-re,
– Módosítsuk a 100-as számot 50-re, és
– Nyomjuk meg a RETURN billentyűt.

– Vigyük be a LIST 80-at,
– Módosítsuk a 32-es számot 16-ra,
– Módosítsuk a 100-as számot 50-re, és
– Nyomjuk meg a RETURN billentyűt.

A módosítások végrehajtása után a RUN paranccsal indítható a program. Más hangzásban és kétszer olyan gyorsan, de a „Let it be" melódia hangzik fel. Egy furulyára emlékeztető hangot fogunk hallani. Ha a 17 és 16 értékeket 33-ra és 32-re visszaállítjuk, újból az eredeti hangzást halljuk. A program ezen részében kizárólag ezek a változások. Vigyük be a „Hey Jude" adatait és RUN paranccsal indítsuk el.

Más lehetőségek is vannak a program módosítására (a számítógépes szakember a módosításra, szerkesztésre editálást mond). A "FOR I = 0 TO IDŐTARTAM * ......" utasítás utáni szám értékének megváltoztatásával változik a zenemű tempója is. Minél kisebb ez a szám, a számítógép annál gyorsabban játssza a művet. A FOR és NEXT utasításokat (70/75-ös és 80/85-ös sorok) késleltető ciklusnak is nevezik. A késleltetés alatt ugyanis a számítógép nullától elszámol egy meghatározott számig. Minél kisebb ez a szám, a számítógép annál előbb készen van a számolással, vagyis a program futási ideje csökken. Ha a számot felezzük, akkor a program kétszer olyan gyorsan fut, ha a számot duplájára növeljük, akkor feleződik a futási sebesség.

A 23-as, 24-es és a 25-ös sorok értékei is változtathatók. A 23-as sorban a dallam hangerősségét szabályozó „alkalmazott" található. A hangerősséget, melynek értékei 1-től 15-ig terjednek, a leghalkabbtól a leghangosabbig be lehet állítani. Rögzítsük le egyszerűen a 23 POKE HANGERŐ, X utasítást, ahol X 0 és 15 közötti érték lehet. Ezt egy hangerőszabályozó gombként is fel lehet fogni; ahol X = 0 esetében a hangerőszabályozó a nullán áll. Az 1, 2, 3 stb. értékekkel nő a hangerősség, melynek maximumát a 15-tel érjük el.

A 24-es és a 25-ös sorokban az úgynevezett ADSR értékeket állítjuk be, melyek pontos leírását a 4. fejezet tartalmazza. Most ne foglalkozzunk sokat ezen értékek jelentésével, ami ráadásul elég bonyolult. Ha kíváncsiságunkat nem tudjuk legyőzni, megpróbálhatunk 0 és 255 közötti értékeket bevinni és rájönni bizonyos törvényszerűségekre.

```
0 REM **** P9. ****
1 :
2 :
3 REM *** "MULL OF KINTYRE"
4 REM
5 REM *** A TAROLOK NULLAZASA
6 REM
7 FOR I=54272 TO 54296
8 POKE I,0
9 NEXT I
10 REM
11 REM *** AZ ALKALMAZOTT JELOLESEK
12 REM
13 ELSO    =54272
14 HANGERO=ELSO+24
15 BE      =ELSO+5
16 KI      =ELSO+6
17 H1      =ELSO
18 H2      =ELSO+1
19 NYOMAS  =ELSO+4
20 REM
21 REM *** A KEZDOERTEKEK BEALLITASA
22 REM
23 POKE HANGERO,15
24 A=0
25 D=6
26 S=8
27 R=9
28 POKE BE,A*16+D
29 POKE KI,S*16+R
30 REM
31 REM *** A HANGFREKVENCIA BEOLVASASA
32 REM
35 DIM FRQ$(59),FL(59),FH(59)
40 FOR I=0 TO 59
41 READ FRQ$(I),FRQ
42 FH(I)=INT(FRQ/256)
43 FL(I)=FRQ-FH(I)*256
44 NEXT I
47 REM
48 REM *** A HANGMAGASSAG ES IDOTARTAM BEOLVASASA
49 REM
50 DIM N1$(100),D1(100)
51 I=0
52 READ N1$(I),D1(I)
53 IF D1(I)>0 THEN I=I+1:GOTO 52
57 REM
58 REM *** A HANGJELEK ATSZAMITASA
59 REM
60 PRINTCHR$(147)CHR$(13)"HANGJEGY-ATSZAMITAS"CHR$(13)
62 DIM L1(100),H1(100)
63 I=0
64 J=59
65 IF FRQ$(J)=N1$(I) THEN L1(I)=FL(J):H1(I)=FH(J):PRINTN1$(I),:GOTO 68
67 J=J-1:GOTO 65
68 I=I+1:IF NOT N1$(I)="" THEN 64
70 REM
```

```
71 REM *** A DARAB LEJATSZASA
72 REM
73 PRINT CHR$(147)
74 PRINT"SEBESSEG (1-9)?"
75 POKE 198,0:WAIT 198,1:GET G$
76 IF VAL(G$)<1 THEN 75
77 SEBESSEG=VAL(G$):KAPCSOLO=0
80 S1=0
81 C1=0
82 POKE H1,L1(S1):POKE H2,H1(S1):POKE NYOMAS,33
84 C1=C1+1:IF C1>(D1(S1)*SEBESSEG/2) THEN POKE NYOMAS,32
85 IF C1<(D1(S1)*SEBESSEG) THEN 90
86 C1=0:S1=S1+1
87 POKE NYOMAS,33
88 POKE H1,L1(S1):POKE H2,H1(S1)
90 IF S1=69 THEN IF C1>(D1(S1)*SEBESSEG-2) THEN KAPCSOLO=1:GOTO 80
91 IF S1=26 THEN IF C1>(D1(S1)*SEBESSEG-2) THEN IF KAPCSOLO=1 THEN 93
92 GOTO 84
93 PRINT:PRINT"MEG EGYSZER? (I)"
94 POKE 198,0:WAIT198,1:GET A$
95 IF A$="I" THEN 73
96 END
97 REM
98 REM *** ADATMEZO
99 REM
100 REM FREKVENCIAK
101 DATA C-2 ,1072
102 DATA C#-2,1136
103 DATA D-2 ,1206
104 DATA D#-2,1275
105 DATA E-2 ,1351
106 DATA F-2 ,1432
107 DATA F#-2,1517
108 DATA G-2 ,1607
109 DATA G#-2,1703
110 DATA A-2 ,1804
111 DATA A#-2,1911
112 DATA B-2 ,2025
121 DATA C-3 ,2145
122 DATA C#-3,2273
123 DATA D-3 ,2408
124 DATA D#-3,2551
125 DATA E-3 ,2703
126 DATA F-3 ,2864
127 DATA F#-3,3034
128 DATA G-3 ,3215
129 DATA G#-3,3406
130 DATA A-3 ,3608
131 DATA A#-3,3823
132 DATA B-3 ,4050
141 DATA C-4 ,4291
142 DATA C#-4,4547
143 DATA D-4 ,4817
144 DATA D#-4,5103
145 DATA E-4 ,5407
146 DATA F-4 ,5728
147 DATA F#-4,6069
148 DATA G-4 ,6430
149 DATA G#-4,6812
150 DATA A-4 ,7217
151 DATA A#-4,7647
152 DATA B-4 ,8101
161 DATA C-5 ,8583
162 DATA C#-5,9094
163 DATA D-5 ,9634
164 DATA D#-5,10207
165 DATA E-5 ,10814
166 DATA F-5 ,11457
167 DATA F#-5,12139
168 DATA G-5 ,12860
169 DATA G#-5,13625
170 DATA A-5 ,14435
171 DATA A#-5,15294
172 DATA B-5 ,16203
181 DATA C-6 ,17167
182 DATA C#-6,18188
183 DATA D-6 ,19269
184 DATA D#-6,20415
185 DATA E-6 ,21629
186 DATA F-6 ,22915
187 DATA F#-6,24278
188 DATA G-6 ,25721
189 DATA G#-6,27251
190 DATA A-6 ,28871
192 DATA B-6 ,32407
200 REM HANGERTEKEK ES IDOTARTAMOK
201 DATA A-4 ,3,B-4 ,1:REM REFREN
202 DATA D-5 ,2,F#-5,4
203 DATA F#-5,2,F#-5,2
204 DATA E-5 ,2,D-5 ,2
205 DATA E-5 ,1,D-5 ,3
206 DATA B-4 ,2,A-4 ,3
207 DATA B-4 ,1,D-5 ,2
```

```
208 DATA F#-5,4,F#-5,2
209 DATA F#-5,2,E-5 ,2
210 DATA D-5 ,2,E-5 ,1
211 DATA D-5 ,3,B-4 ,2
212 DATA A-4 ,3,B-4 ,1
213 DATA D-5 ,1,E-5 ,1
214 DATA D-5 ,9
220 DATA A-4 ,2,D-5 ,2:REM STROFA
221 DATA B-4 ,2,A-4 ,1
222 DATA F#-4,3,A-4 ,2
223 DATA D-5 ,2,F#-5,2
224 DATA E-5 ,2,D-5 ,6
225 DATA B-4 ,2,D-5 ,2
226 DATA C#-5,2,B-4 ,1
227 DATA A-4 ,3,G-4 ,2
228 DATA A-4 ,2,D-5 ,2
229 DATA B-4 ,2,A-4 ,6
230 DATA A-4 ,2,D-5 ,2
231 DATA B-4 ,2,A-4 ,1
232 DATA F#-4,3,A-4 ,2
233 DATA D-5 ,2,F#-5,2
234 DATA E-5 ,2,D-5 ,2
235 DATA E-5 ,2,F#-5,2
236 DATA G-5 ,3,F#-5,1
237 DATA E-5 ,2,D-5 ,1
238 DATA B-4 ,3,B-4 ,2
239 DATA A-4 ,3,B-4 ,1
240 DATA D-5 ,1,E-5 ,1
241 DATA D-5 ,9
299 DATA ,-1
READY.
```

3.3 „Mull of Kintyre" – Egyszerűsített hangjegybevitel

Az előző két mintaprogram, a „Let it be" és a „Hey Jude" bevitele során tapasztalhattuk a rögzítési nehézségeket és a különösen fárasztó DATA sorok bebillentyűzését. Ezt az adatbevitelt a zeneprogramok esetében egyetlen programnyelvnél sem lehet megtakarítani, ugyanis a zenei információkat valamilyen módon meg kell adni. A hangmagassági és hangidőtartam-adatok bevitelének módszere azonban már függ a programozótól. A „Mull of Kintyre" mintaprogram kapcsán új, kényelmesebb módszerrel ismerkedhetünk meg.

A program első pár sora még ismerős lesz, a 30-as sortól azonban lényeges az eltérés. A DATA utasításokkal (100-as sor) kezdődően újabb különbségeket lehet felfedezni az előző mintaprogramokhoz képest.

A módosítás alapötlete az, hogy a hangmagasságok meghatározásához a DATA utasítások esetében négy- és ötjegyű számokat kellett rögzíteni. Ennél egyszerűbb az új adatszerkezet, mellyel a hangmagasság közvetlenül megadható pl.: A–4, B–4, D–5 formában. Ezt mutatja be a „Mull of Kintyre" mintaprogram.

A zenész számára bizonyára ismertebb ez a hangmegadási forma, mellyel viszont újabb probléma merül fel, ugyanis a számítógép közvetlenül ezt az információt nem tudja értelmezni. Ezért a hangjegyeket előbb át kell számítani frekvenciára, amit viszont gyorsabban és pontosabban végez a számítógép. A végeredmény tehát, hogy nem kell rögzíteni a négy- és ötjegyű számokat.

Ezt az átszámító rutint az összes további mintaprogram használni fogja, ezért nem kell újra és újra bebillentyűzni. Nézzük meg részleteiben a program működését.

A „Mull of Kintyre" mintaprogram 100–192-es soraiban találjuk azt a hangfrekvenciát, melyet a melódia minden egyes hangjához hozzá kell rendelni. Mivel a programnak bármikor hozzá kell férnie egy index segítségével a hangfrekvenciához, ezért a 30–44-es sorok között a hangfrekvenciát beírjuk egy adattömbbe. A rutin működését könnyen áttekinthetjük.

A frekvencia beolvasásához három tömböt dimenzionálunk (35-ös sor); egy füzértömböt (FRQ$) a hangok szimbolizálásához, kettőt pedig a frekvencia alsó és felső byte-ban tárolt értékei számára (FL ill. FH). Mivel a DATA utasításokban a frekvenciák 16 bites értékként jelennek meg, ezért egyetlen READ ciklusban egy alsó és egy felső byte beolvasására van szükség, hogy a dallam lejátszása során megtakarítsuk ezt az időigényes folyamatot.

Ez a leírás ugyan bonyolultnak hangzik, egy zenei programozó számára azonban egyszerű és a későbbi programozási munkák során ennek ismerete nem szükséges. Ugyanis a hangjegyek beviteléhez nem kell tudni, hogy azok feldolgozása hogyan zajlik. Pl.: a zongoristának sem kell ismernie a kalapács mechanikus működését a játék során.

A program további részében (50–53-as sorok) egy olyan részprogram található, mely a hangjegyfüzért és a hangjegy időtartamát egy további tömbbe beolvassa. Tulajdonképpeni átszámító rutint a 60–68-as sorok tartalmazzák; a rutin minden hangjegyfüzért összehasonlít a 100–192-es sorokban lévő táblázat füzéreivel.

Ha megtalálta a keresett füzért, akkor a két frekvenciatömbbe (L1, H1) beíródik a választott hangfrekvencia. Hibás bevitel esetén a frekvencia kezdeti értéke (=0) változatlan marad, ezért a kívánt hang a darabban lejátszáskor nem szólal meg. Tehát, ha a dallam nem szólal meg, ellenőrizni kell a DATA utasítások értékeit.

A 73–74-es sorokban felhasználói kérdés található. A képernyőn megjelenik a kérdés, hogy milyen sebességgel játssza le a gép a bevitt dallamot. A megadható értékek 1 és 9 közöttiek lehetnek, ahol az 1-es az egészen gyorsat, a 9-es pedig az egészen lassút jelenti. A közbenső érték választása esetén az 5-ös egy amolyan „normál" sebességet jelent.

A 80–92-es sorok azt a programrészt, lejátszó rutint jelentik, mely a bevitt dallam pontos lejátszását biztosítja. Ez a rutin azonban lényegesen eltér az eddig tárgyaltaktól, hisz a hangjegyeket nem a READ utasítással olvassuk az adatblokkból, mert ezek már az előbb ismertetett három adattömbben találhatók. A rutin úgy van kialakítva, hogy többszólamú művé átalakítható legyen.

A rutin fő elve a számláló (counter) rendszer. A rendszer részletes ismertetését az 5. fejezet fogja tartalmazni. Itt csak annyit, hogy egy változóval (C1) 0-tól kezdve elszámolunk egy, a hang időtartamától függő értékig.

Az eddig bemutatott programok csak lineárisan futtathatók, vagyis ismételt futásuk csak újrahívással lehetséges. Ezt a Commodore BASIC READ-DATA struktúrája okozza, mely csak a tárolt értékek soros feldolgozását teszi lehetővé (azaz egyik értéket a másik után).

A tömb programbeli használatával lehetőség van az ismétlésre, vagyis megvalósítható a refrén – strófa – refrén programfelépítés. Ehhez azonban olyan kapcsolóra (flag) van szükség, ami jelzi, hogy a kívánt zeneműrészletet hányszor kell ismételni.

A példaprogramban egy egyszerű kapcsoló lekérdezési módszert alkalmazunk, ugyanis a programnak csak egyszer kell lejátszania a strófát és a refrént is. A dallam egyszeri lejátszása után tulajdonképpen a DATA sorban elhelyezett adatokat is végigolvastuk, de mivel ezeket egy tömbben tároltuk, a tömb indexének nullára állításával a dallamot akárhányszor újra lejátszhatjuk.

A kapcsoló használatának elmulasztása esetén a darab vég nélkül ismétlődne, mert nem lenne meghatározva a megszakítási feltétel. Ezért a 90-es sorban megjelenik a KAPCSOLO = 1 utasítás a GOTO 80 újraindítási utasítás előtt. A 91-es sorban található a megszakítási feltétel, miszerint ha a kapcsoló értéke 1 és a refrén még egyszer lefutott, akkor vége a darabnak.

A 93–95-ös sorokban egy újabb kérdés található, mely lehetővé teszi a felhasználó számára, hogy ismételten meghallgathassa a művét. Ezzel elkerülhetjük azt, hogy az újrajátszás során a hangjegyátszámítás időtartamát ismételten meg kelljen várni.

Az alábbi „Yesterday" program egy kétszólamú zenemű, mely a „Mull of Kintyre" szerkezete alapján épül fel:

```
0 REM **** P10. ****
1 :
2 :
3 REM *** "YESTERDAY"
4 REM
5 REM *** A TAROLOK NULLAZASA
6 REM
7 FOR I=54272 TO 54296
8 POKE I,0
9 NEXT I
10 REM
11 REM *** AZ ALKALMAZOTT JELOLESEK
12 REM
13 ELSO   =54272
14 HANGERO=ELSO+24
15 BE     =ELSO+5
16 KI     =ELSO+6
17 H1     =ELSO
18 H2     =ELSO+1
19 NYOMAS =ELSO+4
20 REM
```

```
21 REM *** A KEZDOERTEKEK BEALLITASA
22 REM
23 POKE HANGERO,15:POKE ELSO+3,3
24 POKE BE   , 6*16+ 8
25 POKE KI   , 7*16+10
26 POKE BE+7, 2*16+11
27 POKE KI+7,11*16+11
28 W1=64:REM NEGYSZOG
29 W2=32:REM FURESZFOG
30 REM
31 REM *** A HANGFREKVENCIA BEOLVASASA
32 REM
35 DIM FRQ$(59),FL(59),FH(59)
40 FOR I=0 TO 59
41 READ FRQ$(I),FRQ
42 FH(I)=INT(FRQ/256)
43 FL(I)=FRQ-FH(I)*256
44 NEXT I
47 REM
48 REM *** A HANGMAGASSAGOK ES IDOTARTAMOK BEOLVASASA
49 REM
50 DIM N1$(100),D1(100),N2$(100),D2(100)
51 I=0
52 READ N1$(I),D1(I):IF D1(I)>0 THEN I=I+1:GOTO 52
53 I=0
54 READ N2$(I),D2(I):IF D2(I)>0 THEN I=I+1:GOTO 54
57 REM
58 REM *** A HANGJELEK ATSZAMITASA
59 REM
60 PRINTCHR$(147)CHR$(13)"HANGJEGY-ATSZAMITAS"CHR$(13)
61 PRINT,"1. SZOLAM","2. SZOLAM"CHR$(13)
62 DIM L1(100),H1(100),L2(100),H2(100)
63 I=0
64 FOR J=0 TO 59
65 IF FRQ$(J)=N1$(I) THEN L1(I)=FL(J):H1(I)=FH(J):PRINT,N1$(I)
66 IF FRQ$(J)=N2$(I) THEN L2(I)=FL(J):H2(I)=FH(J):PRINT,,N2$(I)
67 NEXT
68 I=I+1:IF NOT (N1$(I)="" AND N2$(I)="") THEN 64
70 REM
71 REM *** A DARAB LEJATSZASA
72 REM
73 PRINT CHR$(147)
74 PRINT"SEBESSEG (1-5)?"
75 POKE 198,0:WAIT 198,1:GET G$
76 IF VAL(G$)<1 OR VAL(G$)>5 THEN 75
77 SEB=VAL(G$):KAPCS=0:GOTO 80
78 FOR X=1 TO SEB*80:NEXT:GOTO 81
79 FOR X=1 TO SEB*80:NEXT
80 S1=0:S2=0
81 C1=0:C2=0
82 POKE H1,L1(S1):POKE H2,H1(S1):POKE NYOMAS,W1 OR 1
83 POKEH1+7,L2(S2):POKEH2+7,H2(S2):POKENYOMAS+7,W2 OR 1
84 C1=C1+1:IFC1>(D1(S1)*SEB/2)THENPOKENYOMAS,W1
85 IFC1<(D1(S1)*SEB)THEN87
86 C1=0:S1=S1+1:POKENYOMAS,W1 OR 1:POKEH1,L1(S1):POKEH2,H1(S1)
87 C2=C2+1:IFC2>(D2(S2)*SEB/2)THENPOKENYOMAS+7,W2
88 IFC2<(D2(S2)*SEB)THEN90
```

```
89 C2=0:S2=S2+1:POKENYOMAS+7,W2 OR 1:POKEH1+7,L2(S2):POKEH2+7,H2(S2)
90 IFS1=28THENIFC1>(D1(S1)*SEB/2)THENIFKAPCS=0THENKAPCS=1:GOTO79
91 IFS1=55THENIFC1>(D1(S1)*SEB/2)ORSEB=1THENIFKAPCS=1THENKAPCS=2:GOTO80
92 IFS1=28THENIFC1>(D1(S1)*SEB/2)THENIFKAPCS=2THENSP=SP+1:S1=56:S2=50:GOTO78
93 IFS1=62THENIFC1>(D1(S1)*SEB/2)THEN95
94 GOTO84
95 PRINT"MEG EGYSZER? (I)":POKE198,0:WAIT198,1:GETA$:IFA$="I"THENGOTO73
96 END
97 REM
98 REM *** ADATMEZO
99 REM
100 REM FREKVENCIAK
101 DATA C-2 ,1072
102 DATA C#-2,1136
103 DATA D-2 ,1206
104 DATA D#-2,1275
105 DATA E-2 ,1351
106 DATA F-2 ,1432
107 DATA F#-2,1517
108 DATA G-2 ,1607
109 DATA G#-2,1703
110 DATA A-2 ,1804
111 DATA A#-2,1911
112 DATA B-2 ,2025
121 DATA C-3 ,2145
122 DATA C#-3,2273
123 DATA D-3 ,2408
124 DATA D#-3,2551
125 DATA E-3 ,2703
126 DATA F-3 ,2864
127 DATA F#-3,3034
128 DATA G-3 ,3215
129 DATA G#-3,3406
130 DATA A-3 ,3608
131 DATA A#-3,3823
132 DATA B-3 ,4050
141 DATA C-4 ,4291
142 DATA C#-4,4547
143 DATA D-4 ,4817
144 DATA D#-4,5103
145 DATA E-4 ,5407
146 DATA F-4 ,5728
147 DATA F#-4,6069
148 DATA G-4 ,6430
149 DATA G#-4,6812
150 DATA A-4 ,7217
151 DATA A#-4,7647
152 DATA B-4 ,8101
161 DATA C-5 ,8583
162 DATA C#-5,9094
163 DATA D-5 ,9634
164 DATA D#-5,10207
165 DATA E-5 ,10814
166 DATA F-5 ,11457
167 DATA F#-5,12139
168 DATA G-5 ,12860
169 DATA G#-5,13625
170 DATA A-5 ,14435
171 DATA A#-5,15294
172 DATA B-5 ,16203
181 DATA C-6 ,17167
182 DATA C#-6,18188
183 DATA D-6 ,19269
184 DATA D#-6,20415
185 DATA E-6 ,21629
186 DATA F-6 ,22915
187 DATA F#-6,24278
188 DATA G-6 ,25721
189 DATA G#-6,27251
190 DATA A-6 ,28871
192 DATA B-6 ,32407
199 REM HANGJEGYEK ES IDOTARTAMOK
200 REM 1. SZOLAM
201 DATA G-4 ,2,F-4 ,1
202 DATA F-4 ,7,A-4 ,1
203 DATA B-4 ,1,C#-5,1
204 DATA D-5 ,1,E-5 ,1
205 DATA F-5 ,1,E-5 ,2
206 DATA D-5 ,1,D-5 ,7
207 DATA D-5 ,1,D-5 ,1
208 DATA C-5 ,1,A#-4,1
209 DATA A-4 ,1,G-4 ,1
210 DATA A#-4,2,A-4 ,1
211 DATA A-4 ,3,G-4 ,2
212 DATA F-4 ,2,A-4 ,1
213 DATA G-4 ,3,D-4 ,2
214 DATA F-4 ,2,A-4 ,1
215 DATA A-4 ,5
220 DATA A-4 ,4,A-4 ,4
221 DATA D-5 ,2,E-5 ,2
222 DATA F-5 ,2,E-5 ,1
223 DATA D-5 ,1,E-5 ,3
224 DATA D-5 ,1,C-5 ,2
225 DATA D-5 ,2,A-4 ,8
226 DATA A-4 ,4,A-4 ,4
227 DATA D-5 ,2,E-5 ,2
228 DATA F-5 ,2,E-5 ,1
229 DATA D-5 ,1,E-5 ,3
230 DATA D-5 ,1,C-5 ,2
231 DATA E-5 ,2,F-5 ,2
232 DATA C-5 ,2,A#-4,2
233 DATA A-4 ,2
240 DATA F-4 ,2,A-4 ,2
```

```
241 DATA G-4 ,2,D-4 .2
242 DATA F-4 ,2,A-4 ,1
243 DATA A-4 ,5
299 DATA ,-1
300 REM 2. SZOLAM
301 DATA F-2 ,2,F-3 ,2
302 DATA F-2 ,2,F-3 ,1
303 DATA F-2 ,1,E-2 ,2
304 DATA E-3 ,2,E-2 ,2
305 DATA E-3 ,1,E-2 ,1
306 DATA D-2 ,2,D-3 ,2
307 DATA D-2 ,2,D-3 ,1
308 DATA D-2 ,1,A#-2,2
309 DATA A#-3,2,C-2 ,2
310 DATA C-3 ,2,F-2 ,6
311 DATA E-2 ,2,D-2 ,2
312 DATA D-3 ,2,C-2 ,2
313 DATA C-3 ,2,A#-2,2
314 DATA F-2 ,1,F-2 ,5
320 DATA C#-3,4,A-2 ,4
321 DATA D-3 ,2,C-3 ,2
322 DATA A#-2,2,G-2 ,2
323 DATA C-3 ,4,C-2 ,4
324 DATA F-2 ,2,A-2 ,2
325 DATA C-3 ,2,A-2 ,1
326 DATA G-2 ,1
327 DATA A-2 ,4,C#-3,4
328 DATA D-3 ,2,C-3 ,2
329 DATA A#-2,2,G-2 ,2
330 DATA C-3 ,4,C-2 ,4
331 DATA F-2 ,8
340 DATA D-3 ,4,C-3 ,4
341 DATA A#-2,2,F-2 ,1
342 DATA F-2 ,5
399 DATA ,-1
READY.
```

3.4. „Yesterday" – Egy kétszólamú zeneprogram

Az eddig bemutatott mintaprogramok csak egyszólamúak voltak, azaz mindig csak a dallam hallatszott, ami viszont eléggé szegényes. A *Yesterday* program már más, összetettebb mint az előzőek.

Lehetőség van a *Mull of Kintyre* program alapján két szólam párhuzamos futtatására. Az ehhez szükséges módosításokat e fejezet keretében ismertetjük.

Az alapvető különbség a két program között az, hogy a *Yesterday* esetében több programsor kétszer fordul elő. (Három szólam esetén bizonyos sorok háromszor fognak megjelenni.) Nézzük például az 52-es és 54-es sorokat! Teljesen egyformák, de az N1$ helyett N2$-t, a D1 helyett pedig D2-t alkalmaztunk. Próbáljunk további hasonló párhuzamokat felfedezni. Először nyilván a 65-ös és 66-os sorok tűnnek fel, a 80-as sortól kezdve azonban a program már eléggé áttekinthetetlen. A hangjegyátszámító rutin még viszonylag egyszerűnek tűnik, de a lejátszó rutin már lényegesen összetettebbé vált (három szólamnál még bonyolultabb).

A lejátszó rutin fő problémája, hogy a két szólam általában egymástól teljesen függetlenül fut. Ami azt eredményezi, hogy a BASIC programozásban előszeretettel használt FOR... NEXT ciklus alkalmazása ebben az esetben lehetetlen.

Ezt a problémát csak a számlálórendszer alkalmazásával lehet megoldani, mely lehetővé teszi az egymástól független két számláló használatát. Így fel lehet dolgozni az egymástól elválasztott különböző hangjegy-időtartamokat.

A 84–89-es sorok tartalmazzák a tényleges lejátszó rutint. Itt a 84-es sor a 87-esnek, a 85-ös a 88-asnak, a 86-os pedig a 89-esnek felel meg. Az első esetben az

L1, H1, D1 tömböket, valamint a W1, C1, S1 változókat, a második esetben pedig az L2, H2, D2 tömböket és a W2, C2, S2 változókat találjuk. Ez biztosítja, hogy mindkét szólamhoz a változók egymástól teljesen függetlenül módosuljanak.
A programfutás vezérlése érdekében itt is található egy kapcsolórendszer, ami viszont bonyolultabb. Ez a kapcsolórendszer valósítja meg a főrész – középrész – főrész – kóda felépítést.

Mi is a kóda? A zenemű bizonyos szakaszát jelölő zenei fogalom, melyet kizárólag a program végén játszanak le és korábban sehol sem fordult elő. Ennek az ellentéte az ún. „úsztatás". Ezt a fajta lezárást leginkább a popzene területén alkalmazzák. A zeneszám befejezésekor ugyanis a refrén újra és újra ismétlődik, közben a hangerő egyre csökken, míg végül már semmit sem lehet hallani. Ebben az esetben a csökkenő dinamikát (hangerőt) alkalmazzuk a lezárásra, de sok esetben a tempó is csökken (ritardando), hogy a mű lecsenghessen.
Ezen folyamat vezérlése során a kapcsoló három értéket vesz fel. A főrészből a középrészre való áttéréskor 0-ról 1-re változik az értéke (90-es sor), visszatéréskor viszont 1-ről 2-re módosul (91-es sor). Ha a főrész befejezése után a kapcsoló értéke 2, akkor a két hangjegyindex a kóda hangjegyeire áll be, lelassul a tempó és elhangzik a kóda. Ezután ismét megjelenik a kérdés, hogy kell-e újra játszani a darabot.

```
0 REM **** P11. ****
1 REM
2 REM
3 REM "STRAWBERRY FIELDS FOREVER"
4 REM THOMAS DACHSEL FELDOLGOZASA
5 REM
6 REM
7 FOR I=54272 TO 54296
8 POKE I,0
9 NEXT I
10 REM
11 REM A KEZDOERTEKEK BEALLITASA
12 REM
13 ELSO=54272
14 TL=ELSO
15 TH=ELSO+1
16 PL=ELSO+2
17 PH=ELSO+3
18 WE=ELSO+4
19 AD=ELSO+5
20 SR=ELSO+6
21 LF=ELSO+21
22 HF=ELSO+22
23 RE=ELSO+23
24 LA=ELSO+24
27 REM
28 REM A FREKVENCIAK BEOLVASASA
29 REM
30 DIM FR$(11),FL(11,4),FH(11,4)
31 FOR I=0 TO 11
32 READ FR$(I)
```

```
33 FOR J=0 TO 4
34 READ FR
35 FH(I,J)=INT(FR/256)
36 FL(I,J)=FR-FH(I,J)*256
37 NEXT J
38 NEXT I
47 REM
48 REM A HANGJEGYADATOK BEOLVASASA
49 REM
50 DIMN1$(200),O1(200),D1(200)
51 DIMN2$(200),O2(200),D2(200)
52 DIMN3$(200),O3(200),D3(200)
53 I=0
54 READN1$(I),O1(I),D1(I):IFD1(I)>0THENI=I+1:GOTO54
55 I=0
56 READN2$(I),O2(I),D2(I):IFD2(I)>0THENI=I+1:GOTO56
57 I=0
58 READN3$(I),O3(I),D3(I):IFD3(I)>0THENI='+1:GOTO58
59 REM
60 REM AZ ATSZAMITO RUTIN
61 REM
62 DIML1(200),H1(200),L2(200),H2(200),L3(200),H3(200)
63 I=0:PRINTCHR$(147)"1. SZOLAM:"
64 FOR J=0 TO 11
65 IFFR$(J)=N1$(I)THENL1(I)=FL(J,O1(I)-2):H1(I)=FH(J,O1(I)-2):PRINTN1$(I)
66 NEXT
67 I=I+1:IFD1(I)>0THEN64
68 I=0:PRINT:PRINT"2. SZOLAM:"
69 FOR J=0 TO 11
70 IFFR$(J)=N2$(I)THENL2(I)=FL(J,O2(I)-2):H2(I)=FH(J,O2(I)-2):PRINTN2$(I)
71 NEXT
72 I=I+1:IFD2(I)>0THEN69
73 I=0:PRINT:PRINT"3. SZOLAM:"
74 FOR J=0 TO 11
75 IFFR$(J)=N3$(I)THENL3(I)=FL(J,O3(I)-2):H3(I)=FH(J,O3(I)-2):PRINTN3$(I)
76 NEXT
77 I=I+1:IFD3(I)>0THEN74
87 REM
88 REM A KEZDOHANGOK BEALLITASA
89 REM
90 POKEAD,41:POKEAD+7,41:POKEAD+14,41
91 POKESR,121:POKESR+7,121:POKESR+14,121
92 POKELA,31:POKERE,0
93 W1=16:W2=16:W3=16
94 KAPCS=0
95 S1=0:S2=0:S3=0:REM MUTATOK
96 C1=0:C2=0:C3=0:REM SZAMLALO
100 REM
101 REM A DARAB LEJATSZASA
102 REM
103 POKETL,L1(S1):POKETH,H1(S1):POKEWE,W1 OR 1
104 POKETL+7,L2(S2):POKETH+7,H2(S2):POKEWE+7,W2 OR 1
105 POKETL+14,L3(S3):POKETH+14,H3(S3):POKEWE+14,W3 OR 1
110 C1=C1+1:IFC1>(D1(S1)*.5)THENPOKEWE,W1
111 IFC1<(D1(S1))THEN120
112 C1=0:S1=S1+1:POKETL,L1(S1):POKETH,H1(S1):POKEWE,W1 OR 1
120 C2=C2+1:IFC2>(D2(S2)*.7)THENPOKEWE+7,W2
```

```
121 IFC2<(D2(S2))THEN130
122 C2=0:S2=S2+1:POKETL+7,L2(S2):POKETH+7,H2(S2):POKEWE+7,W2 OR 1
130 C3=C3+1:IFC3>(D3(S3)*.8)THENPOKEWE+14,W3
131 IFC3<(D3(S3))THEN140
132 C3=0:S3=S3+1:POKETL+14,L3(S3):POKETH+14,H3(S3):POKEWE+14,W3 OR 1
140 IFS1=14THENW1=64:POKEPH,3
141 IFS2=2THENIFC2=9THENW2=32:W3=32:POKEAD+14,7:POKESR+14,169:POKESR+7,139
142 IFS2=22THENIFC2=5THENPOKEHF,210:POKERE,7:IFKAPCS=2THEN151
143 IFS1=96THENIFC1=15THENPOKERE,0:KAPCS=KAPCS+1:S1=14:S2=3:S3=9:GOTO96
150 IFD1(S1)>0ORD2(S2)>0ORD3(S3)>0THEN110
151 POKEWE,0
152 PRINTCHR$(147)"MEG EGYSZER? (I)":POKE198,0:WAIT198,1
153 GETC$:IFC$<>"I"THENPOKEWE,0:END
154 C1=0:C2=0:C3=0:S1=0:S2=0:S3=0:GOTO90
200 REM A FREKVENCIAK
201 DATA C ,1072,2145,4291, 8583,17167
202 DATA C#,1136,2273,4547, 9094,18188
203 DATA D ,1206,2408,4817, 9634,19269
204 DATA D#,1275,2551,5103,10207,20415
205 DATA E ,1351,2703,5407,10814,21629
206 DATA F ,1432,2864,5728,11457,22915
207 DATA F#,1517,3034,6069,12139,24278
208 DATA G ,1607,3215,6430,12860,25721
209 DATA G#,1703,3406,6812,13625,27251
210 DATA A ,1804,3608,7217,14435,28871
211 DATA A#,1911,3823,7647,15294,30588
212 DATA B ,2025,4050,8101,16203,32407
1000 REM 1. SZOLAM
1001 DATA B ,4,5
1002 DATA B ,4,5
1003 DATA B ,4,5
1004 DATA B ,4,5
1005 DATA B ,4,5
1006 DATA B ,4,5
1007 DATA A ,4,5
1008 DATA G#,4,7
1009 DATA F#,4,3
1010 DATA A ,4,6
1011 DATA E ,4,3
1012 DATA G#,4,4
1013 DATA F#,4,5
1014 DATA E ,4,10
1015 DATA   ,0,4
1016 DATA C#,5,2
1017 DATA C#,5,2
1018 DATA D ,5,2
1019 DATA C#,5,4
1020 DATA A ,4,6
1021 DATA E ,4,2
1022 DATA F#,4,2
1023 DATA B ,4,2
1024 DATA A ,4,4
1025 DATA G ,4,7
1026 DATA G ,4,3
1027 DATA A ,4,3
1028 DATA B ,4,3
1029 DATA E ,4,8
1030 DATA   ,0,15
1031 DATA G ,4,3
1032 DATA A ,4,3
1033 DATA A#,4,3
1034 DATA E ,4,8
1035 DATA   ,0,6
1036 DATA E ,4,2
1037 DATA E ,5,2
1038 DATA D ,5,2
1039 DATA C#,5,2
1040 DATA B ,4,2
1041 DATA A#,4,1
1042 DATA B ,4,1
1043 DATA C#,5,4
1044 DATA   ,0,10
1045 DATA C#,5,2
1046 DATA A ,4,2
1047 DATA F#,4,2
1048 DATA C#,5,2
1049 DATA A ,4,2
1050 DATA E ,4,2
1051 DATA B ,4,2
1052 DATA A ,4,10
1053 DATA   ,0,6
1054 DATA B ,4,3
1055 DATA B ,4,3
1056 DATA B ,4,3
1057 DATA B ,4,3
1058 DATA B ,4,3
1059 DATA B ,4,3
```

```
1060 DATA B ,4,6
1061 DATA B ,4,11
1062 DATA   ,0,2
1063 DATA A ,4,2
1064 DATA A ,4,2
1065 DATA A ,4,2
1066 DATA C#,5,2
1067 DATA B ,4,2
1068 DATA A ,4,2
1069 DATA G#,4,2
1070 DATA A ,4,1
1071 DATA G#,4,1
1072 DATA F#,4,4
1073 DATA   ,0,12
1074 DATA A ,4,2
1075 DATA A ,4,2
1076 DATA A ,4,2
1077 DATA G#,4,3
1078 DATA F#,4,1
1079 DATA E ,4,2
1080 DATA C#,5,2
1081 DATA A ,4,2
1082 DATA A ,4,1
1083 DATA A ,4,1
1084 DATA A ,4,1
1085 DATA B ,4,1
1086 DATA A ,4,1
1087 DATA G#,4,1
1088 DATA F#,4,4
1089 DATA   ,0,6
1090 DATA A ,4,2
1091 DATA A ,4,2
1092 DATA A ,4,2
1093 DATA B ,4,2
1094 DATA A ,4,2
1095 DATA G#,4,2
1096 DATA A ,4,2
1097 DATA A ,4,16
1999 DATA ,-1,-1
2000 REM 2. SZOLAM
2001 DATA   ,0,58
2002 DATA D ,4,5
2003 DATA E ,4,10
2004 DATA A ,3,32
2005 DATA B ,3,8
2006 DATA B ,3,8
2007 DATA B ,3,8
2008 DATA   ,0,8
2010 DATA C#,3,2
2011 DATA F#,3,2
2012 DATA A#,3,12
2013 DATA A#,3,16
2014 DATA F#,3,6
2015 DATA G#,3,2
2016 DATA A#,3,8
2017 DATA A#,4,2
2018 DATA F#,4,2
2019 DATA E ,4,2
2020 DATA F#,4,2
2021 DATA F#,3,6
2022 DATA F#,3,6
2023 DATA C#,3,12
2024 DATA   ,0,6
2025 DATA G#,4,9
2026 DATA G#,4,9
2027 DATA G#,4,9
2028 DATA A ,4,4
2029 DATA G#,4,4
2030 DATA F#,4,8
2031 DATA E ,4,8
2032 DATA D ,4,8
2033 DATA F#,3,4
2034 DATA E ,3,4
2035 DATA F#,4,8
2036 DATA D ,4,6
2037 DATA E ,4,4
2038 DATA C#,4,6
2039 DATA A ,3,4
2040 DATA   ,0,4
2041 DATA F#,4,8
2042 DATA E ,4,8
2043 DATA F#,4,8
2044 DATA C#,4,8
2999 DATA ,-1,-1
3000 REM 3. SZOLAM
3001 DATA E ,4,10
3002 DATA D#,4,10
3003 DATA D ,4,10
3004 DATA F#,4,5
3005 DATA F ,4,5
3006 DATA C#,4,9
3007 DATA B ,3,9
3008 DATA A ,3,5
3009 DATA C#,4,10
3010 DATA A ,2,32
3011 DATA A ,2,6
3012 DATA E ,2,2
3013 DATA E ,2,8
3014 DATA E ,2,2
3015 DATA G ,2,2
3016 DATA B ,2,2
3017 DATA G ,2,2
3018 DATA E ,2,2
3019 DATA G ,2,2
3020 DATA B ,2,2
3021 DATA G ,2,2
3022 DATA F#,2,16
3023 DATA F#,2,6
3024 DATA F#,2,2
3025 DATA F#,2,8
3026 DATA D ,2,6
3027 DATA E ,2,2
3028 DATA F#,3,8
3029 DATA F#,3,6
```

```
3030 DATA E ,3,2
3031 DATA D ,3,6
3032 DATA D ,3,4
3033 DATA E ,3,2
3034 DATA A ,2,18
3035 DATA E ,4,9
3036 DATA D#,4,9
3037 DATA D ,4,9
3038 DATA F#,4,4
3039 DATA F ,4,4
3040 DATA C#,4,16
3041 DATA   ,0,8
3042 DATA F#,2,4
3043 DATA E ,2,4
3044 DATA D ,2,8
3045 DATA E ,2,8
3046 DATA A ,2,8
3047 DATA F#,2,4
3048 DATA E ,2,4
3049 DATA D ,2,8
3050 DATA E ,2,8
3051 DATA D ,2,8
3052 DATA A ,2,8
3999 DATA ,-1,-1
READY.
```

3.5. Egy háromszólamú BASIC zenei program szerkezete

A „Strawberry Fields Forever" program felépítése a BASIC nyelv zenei lehetőségének felső határát képviseli. A korábbi bemutatott mintaprogramok esetében lehetőség nyílt a lejátszás sebességének változtatására, vagyis rendelkezésre állt bizonyos sebességtartalék. A nem lineáris lekérdezéssel írt háromszólamú program esetében azonban ilyen lehetőség nincs. (Mint pl. a strófa – refrén – strófa – refrén esetén.)

A többszólamú programozás esetében a Commodore BASIC futása a tömböket feldolgozó utasítások, az IF–THEN elágazások összekapcsolásával igen lassú. Mivel ezek alkalmazása programozástechnikailag szükségszerű, minden hangjegy után megtörténik, ami nagyarányú sebességkorlátozással jár.

Mindezek ellenére lehet a BASIC-ben háromszólamú művet programozni. A programozás mikéntjét a következőkben látni fogjuk.
A program billentyűzetről való bevitele során tapasztalni fogjuk, hogy a háromszólamú zenei program viszonylag hosszú. Az ilyen jellegű programok beírása során ügyelni kell

- a program áttekinthetőségének megtartására, és
- ahol csak lehet, az időmegtakarításra.

A gyakorlott BASIC programozók már bizonyára ismernek néhány trükköt.
Nézzük előbb az áttekinthetőséget.

Az áttekinthetőség a Commodore BASIC alkalmazása esetén igen kényes kérdés, mert ez szembenáll a memóriamegtakarítással. Ugyanis minden egyes szóközkarakter lefoglal egy memóriahelyet, így növeli a program hosszát és lassítja futását. A „Strawberry Fields Forever" program 110–132-es soraiban egyetlen szóköz sem található. A programozók számára ez természetes, de az informatikai szakembernek szokatlan. Ellentétben a 31–38-as sorokkal, ahol a rendező ciklus és a bekezdések egymásba ágyazása ugyan szemléletes, de a szóközök hosszabbá és lassúbbá teszik a programot.

Ha a programot a listában található módon akarjuk bevinni, akkor rendkívül nagy nehézségekre kell felkészülnünk. Ugyanis a normál soreleji szóközöket az editor figyelmen kívül hagyja. Ha ugyanez a SHIFT-elt szóközkarakterrel (space) történik közvetlenül a sorszám után, akkor ilyen nehézségekkel nem kell számolni.

Az áttekinthetőséget növeli a REM utasítás, azonban helyszűke miatt spórolni kell vele, mert maga a sorszám és a REM utasítás öt byte területet foglal le, eltekintve a részletes kommentártól.
Programbevitel esetén az időnyerés általában együtt jár a memóriahely megtakarítással is. Nézzük például a 13–24-es sorokat. A korábbi programoknál minden esetben kiírásra került az ELSŐ=54272 kifejezés. Mint ismeretes, a Commodore BASIC esetében a változók azonosítására két betű használata elegendő, a többi betűt a feldolgozás során az értelmező (interpreter) nem veszi figyelembe. Példánk esetében is elegendő az EL használata az ELSŐ szó helyett. A program számára tehát a FALI és a FAGYI azonosító szavak ugyanazt jelentik. Ez az eset is jól szemlélteti, hogy az áttekinthetőség a memóriatakarékosság rovására mehet.

A program billentyűzetről történő bevitelekor hol nyerhetünk még időt? Végig kell nézni a programot, hogy vannak-e szinte teljesen azonos vagy csak néhány jelben eltérő sorok.
A „Strawberry Fields Forever" programban az 50-es, 51-es és 52-es sorok három szám kivételével teljesen azonosak. A sorok beírásakor az alábbiak szerint járhatunk el. Beírjuk a lista szerint a 50 DIMN1$(200),01(200),D1(200) sort, majd megnyomjuk a RETURN-t. A következő lépésben ugorjunk vissza a kurzorral az előzőleg bevitt sor kezdő karakterére. Ezután a „kurzor jobbra" billentyűvel módosíthatjuk a változó karaktereket. Ezzel az eljárással tehát csak az eltérő jelek bevitelére van szükség. Példánkban csak a sorszámot kell 50-ről 51-re és három 1-est 2-esre módosítani. A RETURN billentyű lenyomása után mindkét sor bekerül a memóriába. Ezután az 51-es sorból ugyanígy elkészíthetjük az 52-est.
Nem kell azonban mindig egyenként módosítani a sorokat, mert az eljárás tobb sorra is kibővíthető. Nézzük pl. az 53-as és 54-es sorokat. Ezeket a sorokat bevisszük egymás után úgy, hogy közvetlenül egymás alá essenek, majd a kurzorral visszaléptetünk az 53-as sor első karakterére. Mivel az 53-as és az 55-ös sorok megegyeznek, ezért csak a sorszámot kell 55-re módosítani és a RETURN billentyű megnyomásával a sor beíródik a memóriába. A második sorban a kurzor jobbra billentyű segítségével négy 1-est 2-esre kell változtatni, majd a GOTO 54-et GOTO 56-ra kell átírni. Az 57-es, ill. az 58-a sorok bevitele esetében ugyanezen eljárást kell követni.

Az eljárás megfelelő elsajátítása és a kurzorvezérlő billentyűk kezelésében szerzett gyakorlat után lényegesen több időt lehet megtakarítani (a Commodore 64-es egyik hátránya, hogy a kurzor balra és felfelé történő mozgatásához a SHIFT gombot minden esetben használni kell).

Idővel automatikussá válik azon sorok felismerése, ahol a fenti eljárást alkalmazni lehet. A programban hasonló módon rögzíthetők még a 63–67-es soroknak a 68–72-esre és a 73–77-esre, valamint a 110–112-es soroknak a 120–122-esre és a 130–132-esre történő átalakításával.
További időmegtakarítás, ha ezeket a hasonló sorokat egy BASIC-TOOLKIT segítségével AUTO utasítással visszük be. A BASIC-TOOLKIT a programozási munkát elősegítő programcsomag.*
A 200-as sortól kezdődően szükség van ilyen programozói megoldásra a sok DATA utasítás beviteléhez. Ez esetben is lehetőség van a kurzorvezérlő billentyűk használatára, azonban a sorszámot minden esetben meg kell változtatni, ami egy ideig nagyon unalmas. BASIC-TOOLKIT segítségével ez sokkal egyszerűbb, mert például átírhatjuk az egyik funkcióbillentyűt a DATA utasításra. Legyen ez pl. az F1 billentyű. Majd kiadjuk az AUTO 200,1 utasítást és elindítjuk a programot. Egy sor elkészülte után megnyomjuk a RETURN és az F1 billentyűket. Ennek hatására megjelenik a következő sorszám és a DATA utasítás.

Miután elég részletesen megbeszéltük a háromszólamú „Strawberry Fields Forever" program bevitelét, térjünk rá a szerkezetének bemutatására.
A „Yesterday" program alapos tanulmányozása után nem okoz gondot az ott alkalmazott programszerkezet három szólamra történő átültetése. A „Strawberry Fiedls Forever" program bevitele során láthattuk, hogy vannak olyan sorok, amelyek szinte teljesen azonosak. Ezek a sorok minden esetben hármas csoportokban találhatók, mivel három szólamú a zenemű. A példa mutatja, hogy azon zeneprogramnak, amely három szólamot vezérel, mennyivel több „dolga" van a kevesebb szólamúéhoz képest. A memóriaigénye is legalább háromszor annyi, nem elsősorban a program (mind a három szólamnak megvan a hangmagassága és időtartama), hanem az általa használt tömbök miatt.

Figyelembe kell venni azonban a tömbök kezelhetőségének a maximumát. A példában 15 tömb található összesen 200 elemmel, mely elemek egyenként 5 byte-ot foglalnak le. Az összes igénybe vett memóriaterület 15 000 byte, aminek a kezelése a BASIC-értelmező számára már nehézkes.

Mindezek ellenére az elsődleges célunk a rendelkezésre álló memória igénybevételének csökkentése, illetve a meglévő helyek maximális kihasználása. A hangfrekvencia esetében például ez történt. A „Yesterday" programban minden hangfrekvenciának saját sorszáma volt a 100-astól kezdődően 192-ig.

A mostani példaprogramunkban az összes hangfrekvencia sűrűn egymás mögött a 201–212-es sorokban található, mely memóriamegtakarítást jelent, figyelembe véve azt, hogy minden sorszám a DATA utasítással együtt 5 byte memóriahelyet foglal el. Ezen túlmenően az oktávok (C2, C3, C4 stb.) nem hangjegyfüzérekben

* Magyarországon a HELP + segédprogram az elterjedtebb. Az AUTO utasítás megfelel a #G HELP + parancsnak. (A fordító megjegyzése.)

vannak megadva, hanem indexként. Tehát a hangfrekvenciának két dimenziója van, egyik a tizenkét félhangra, másik az öt lehetséges oktávra.

A program szerkezete röviden:

30–38-as sorok A DATA utasításokkal megadott hangfrekvenciák beolvasása a megfelelő tömbbe.

50–58-as sorok A hangmagasságok és időtartamok beolvasása mindhárom szólam számára.

62–77-es sorok A tényleges frekvenciák meghatározása a hangmagasságokból egy átszámító rutin segítségével.

90-es sor E sortól kezdődően egy háromszólamú lejátszó rutin található.

A rutint egy hármas számláló rendszer vezérli. Minden szólamnak van külön számlálója (C1, C2 és C3) és indexe (S1, S2 és S3), a szólamokénti három tömbhöz (L1, H1 és D1; L2, H2 és D2; L3, H3 és D3) úgy, hogy a három szólam időtartamát egymástól függetlenül lehet vezérelni.

A kapcsolóvezérlő rendszer a 140–143-as sorokban található, mely lehetővé teszi a hangszín szabályozását is. A lágy, furulyaszerű hangzású (háromszög hullámforma) bevezető után a kapcsoló átkapcsol egy csengő hangzásra. A mű fűrésze után egy tompább, lágyabb hangzás hallatszik (aluláteresztő szűrő), mely a közbenső részhez tartozik.
Egy kapcsoló vezérli a mű teljes folyamatát, mely a következő: Bevezetés – fűrész – közjáték – fűrész – közjáték – fűrész. A 94-es sorban a kapcsoló 0 kezdőértéket kap. A közjáték vége után értéke minden esetben eggyel növekszik (143-as sor). A fűrész vége után a program ellenőrzi, hogy a kapcsoló értéke elérte-e a kettőt (142-es sor). Igen válasz esetén a mű befejeződik.

3.6. Útmutató saját zeneprogram írásához

A „Strawberry Fields Forever" program kellő tanulmányozás után úgy módosítható, hogy az tetszés szerinti zenét játsszon. Az alábbiakban ezen módosítás folyamatát írjuk le:

Fix programsoroknak nevezzük azokat a sorokat, melyek beépítése minden esetben kötelező. Azokat a sorokat pedig, melyek műről műre változnak, *változó* soroknak nevezzük. Nézzük a programot:

0–89 A DIM utasításokat tartalmazó sorok kivételével folyamatosan fix sorokkal találkozunk. A dimenzió-indexek növelésére bizonyos körülmények között szükség lehet, de a nagysága befolyásolja a program hosszát. Az esetleges módosításoknál ügyelni kell azonban arra, hogy az indexek az összetartozó tömböknél megegyezzenek.

Összetartozó tömbök az N1$, O1, D1, L1 és H1; az N2$, O2, D2, L2 és H2; valamint az N3$, O3, D3, L3 és H3.

90–93 Ezek a sorok változóak, mert itt történik a három szólam kezdő hangszínének beállítása. Azzal, hogy ide milyen értékek írhatók, a 4. fejezet foglalkozik.

94–132 A sorok fixek, a lejátszó rutint tartalmazzák.

140–149 Ezek változó sorok, ahol különféle kapcsoló lekérdezéseket lehet programozni. A 140-es sorra akkor is szükség van, ha nincs kapcsoló lekérdezés (pl.: 140 REM), mivel a 131-es sor erre a sorszámra ugrik.

150–212 A sorok fixek. A darab megismétlésére vonatkozó kérdést és a frekvenciatáblázatot tartalmazzák.

Az 1000–1998, a 2000–2998 és a 3000–3998-as sorszámok közötti programsorok az egyes szólamok hangfrekvenciáinak és hangidőtartamainak megfelelően szabadon változtathatók. Lehetőség van továbbá arra is, hogy több hang adatai egy sorba kerüljenek. Egy hanghoz ezáltal mindig három, egymástól vesszővel elválasztott érték tartozik, melyek az alábbiak:

Az első érték egy füzér, amely a játszandó félhangot szimbolizálja (C, C#, D, D#, E, F, F#, G, G#, A, A#, B). De üres füzér is állhat az első értékként (mivel READ esetén a DATA "", ... vagy a DATA, ... megadások üres füzért eredményeznek). Ebben az esetben a frekvencia 0 a megadott időtartamig (a harmadik érték). Ezt a megoldást szünet jelölésére lehet használni, mert semmi sem hallható.

A második érték az oktáv, melynek értéke 2 és 6 közé kell essen.

A harmadik érték az időtartam, ahol az 1-es egy nyolcadot, a 2-es egy negyedet, a 3-as egy pontozott negyedet, a 4-es egy félhangot jelöl stb.

Az 1999-es, a 2999-es és a 3999-es sorok fixek, mert az egyes szólamok hangmagassági és időtartamadatait választják el. Ha az 54-es (ill. az 56-os és 58-as) sor 0 vagy negatív értéket olvas be, akkor ezt a program úgy értelmezi, hogy a kérdéses szólam valamennyi hangját beolvasta. A –1 ebben a programban egy határoló számnak felel meg, amely megmutatja, hogy hol van az adat.

3.7. Megjegyzések a mintaprogramhoz

Miért a Beatles dalai kerültek mintaprogramként a könyvbe? Mint ismeretes, a Commodore 64-es hang chipje többre képes annál, hogy „csak" Bach fúgákat játsszon vagy hangeffektusokat állítson elő. Ha eltekintünk a „csak" három szólamtól, akkor gyakorlatilag a zene valamennyi területét felölelhetjük, így a pop- és a rockzenét is. Ezért kiváló példák egy olyan együttes dalai, melyeket még ma is játszanak, pedig a hatvanas évek zenéjét határozták meg elsősorban. Ez az együttes a Beatles.

John Lennon, Paul McCartney, George Harrison és Ringo Starr olyan nevek, melyeket az egész világ ismer. Már olyan sok könyv látott napvilágot a Beatles mítoszról, hogy jelenleg karrierjük csak néhány állomásáról írunk.

Egy város nevét mindig egy levegővétellel említik a Beatles-szel, az angliai Liverpoolt, ahonnan a Beatles fiúk származnak. Karrierjük első fontos állomása a hamburgi Star Club volt. 1963: A Beatles éve. A legjelentősebb, ebben az évben kiadott dalok a „Love Me Do", a „Please Please Me" és a leginkább ismert „She Loves You" voltak. 1963–1964-ben az egész világot magával ragadta, az ún. „Beatlemánia". Az együttes ezen idő alatt milliószám adta el nagy- és kislemezeit. A megjelenő filmek, a különböző turnék, az együttes botrányai és még sok minden más növelték egyre jobban a hírnevüket.

A szórakoztatóipar hosszú időn keresztüli kézbentartására a Beatles sem volt képes. 1966 után nem léptek fel nyilvánosan, kizárólag lemez stúdiókban dolgoztak. Személyi és gazdasági problémák jelenkeztek akkor, amikor az együttes a saját menedzselését szerette volna átvenni. Ezek a problémák a kiváltó okai, hogy a Beatles 1970-ben feloszlott. A továbbiakban mind a négyen egyéni elképzelések alapján próbáltak továbbhaladni és sikerük is különböző volt.

Azok az álhírek, hogy a Beatles újra fellép, a nyolcvanas évek elején végleg megszűntek, amikor John Lennont New Yorkban egy rajongó lelőtte. Az együttesből egyedül Paul McCartney az, aki időről időre a mai napig is sikeres.

Az örökzöld dalok nagy része a Lennon–McCartney szerzőpárostól származik. A „Strawberry Fields Forever" John Lennon, a „Yesterday" és a „Let it be" Paul McCartney szerzeménye.

A Beatles dalainak nagyszámú feldolgozása létezik, melyekből a szerzői jogdíj még mindig milliókra rúg. Szimfonikus zenekarok, szórakoztató együttesek és más számtalan pop- és rockegyüttes (The Beatles Revival Band) játssza dalaikat. Az ismertebb dallamok a Commodore 64-esre is megjelentek. A könyv áttanulmányozása során talán akad majd valaki, aki folytatja ezt a hagyományt.

4. A COMMODORE 64-ES HANGZÁS REGISZTEREI

4.1. Általános tudnivalók

Előzőleg a hang chipet úgy ábrázoltuk, mint egy irodaház, melyben sok szoba van és ezekben tisztviselők ülnek. A zeneprogramot pedig kézbesítőként kezeltük, aki meghatározott sorrendben felkeresi a szobákat, és a tisztviselőknek új rendelkezést ad. A továbbiakban azzal foglalkozunk, hogy milyen hivatalnokok léteznek és mi a feladatuk.

A Commodore 64-es hang chip „irodájának" 29 szobájában 215 alkalmazott található. A szobák sorszáma 54 272-től 54 300-ig fut. Egyszerűsítés után ezeknek a számozása 0-tól 28-ig tart, és kellően nagynak kell elképzelni ezen szobákat.

Továbbá minden helyiségben nyolc íróasztal található, melyek számozása 0-tól 7-ig terjed. Az íróasztalok mögött ülnek az alkalmazottak. Ha egy meghatározott alkalmazotthoz kívánunk fordulni, akkor a szobaszám és az íróasztal számának a megadása szükséges.

Tulajdonképpen mit is mondhatunk ezeknek az alkalmazottaknak? Ilyenkor nem szabad megfeledkezni arról, hogy valójában egy számítógéppel van dolgunk, mely csak a nullákat és az egyeseket érti.

Ez a helyzet az alkalmazottakkal is, hisz ők is csak a 0-t és az 1-et értik. Ez a tudás önmagában egy hang lejátszásához kevés, ezért egyszerre több alkalmazott együttes „munkájára" van szükség.

A probléma csak az, hogy a hang chip közvetlenül nem érti a nullákat és az egyeseket. Tehát íly módon nem érhetők el az alkalmazottak. Az elérést a BASIC biztosítja, azonban nem nullákkal és egyesekkel, hanem decimális számokkal.

A BASIC-ben van egy olyan utasítás, mely lehetővé teszi az alkalmazottak elérését. Az utasításban az A a kívánt szoba száma, az X pedig a nullák vagy egyesek helye. Az X=0 esetében a teljes tagot, X=1 esetében az X-et el lehet hagyni.

```
POKE A, X*1+X*2+X*4+X*8+X*16+X*32+X*64+X*128

         0   1   2   3   4    5    6    7
```

Az X-ek alatt lévő számok az elérni kívánt alkalmazottakat jelölik. Pontosabban, mivel a 2 hatványaival van dolgunk, az eredmény nem változna, ha az aritmetikai műveletet (+) logikai művelettel (OR) helyettesítenénk.

A képlet megjegyzése azonban körülményes, ezért egyszerűbb, ha néhány állandó jellegű értéket rögzítünk. A POKE A,0 utasítással valamennyi alkalmazott 0 értéket kap. Ha ugyanezen alkalmazottaknak 1 értéket kívánunk adni, akkor a POKE A,255 utasítást kell alkalmazni.

A hang chip különböző funkciói szándékosan nincsenek az egyes alkalmazottak között felosztva. A legtöbb szobában az alkalmazottak szorosan együttműködve dolgoznak, tehát egyszerre kell őket megszólítani.

Nézzük meg, hogyan dolgoznak együtt az alkalmazottak. Ehhez azonban nem szükséges egy komplett BASIC program, mert néhány dolgot csak egyszer akarunk kipróbálni.

A példa szemléltetése érdekében és a tárolóban lévő BASIC program véletlenszerű módosítása nélkül az alábbi utasításokat közvetlenül bebillentyűzhetjük.

A program az ELSŐ = 54272 utasítással kezdődik.
Ezután a HANGERŐ = ELSŐ + 24 és a POKE HANGERŐ, 15 utasítások következnek. Az utóbbi utasítás a hang chip „hangerő gombját forgatja", melyet bizonyára már ismerünk.

Ha jól odafigyelünk, akkor hangot ugyan még nem, de egy kattanást azért hallunk a hangszóróból. Először meg kell adnunk, hogy milyen hangokat szeretnénk hallani. Segédeszközként egy frekvenciatáblázat áll rendelkezésünkre:

4.2. Hangfrekvencia táblázat

Hangjegy	Decimális	Alsó byte	Felső byte	Hexadecimális	Alsó byte	Felső byte
C–0	268	12	1	$010C	$OC	$01
C#–0	284	28	1	$011C	$1C	$01
D–0	301	45	1	$012D	$2D	$01
D#–0	318	62	1	$013E	$3E	$01
E–0	337	71	1	$0151	$51	$01
F–0	358	102	1	$0166	$66	$01
F#–0	379	123	1	$017B	$7B	$01
G–0	401	145	1	$0191	$91	$01
G#–0	425	169	1	$01A9	$A9	$01
A–0	451	195	1	$01C3	$C3	$01

Hangjegy	Decimális	Alsó byte	Felső byte	Hexadecimális	Alsó byte	Felső byte
B–0	477	221	1	$01DD	$DD	$01
H–0	506	250	1	$01FA	$FA	$01
C–1	536	24	2	$0218	$18	$02
C#–1	568	56	2	$0238	$38	$02
D–1	602	90	2	$025A	$5A	$02
D#–1	637	125	2	$027D	$7D	$02
E–1	675	163	2	$02A3	$A3	$02
F–1	716	204	2	$02CC	$CC	$02
F#–1	758	246	2	$02F6	$F6	$02
G–1	803	35	3	$0322	$23	$03
G#–1	851	83	3	$0353	$53	$03
A–1	902	134	3	$0386	$86	$03
B–1	955	187	3	$03BB	$BB	$03
H–1	1012	244	3	$03F4	$F4	$03
C–2	1072	48	4	$0430	$30	$04
C#–2	1136	112	4	$0470	$70	$04
D–2	1204	180	4	$04B4	$B4	$04
D#–2	1275	251	4	$04FB	$FB	$04
E–2	1351	71	5	$0547	$47	$05
F–2	1432	152	5	$0598	$98	$05
F#–2	1517	237	5	$05ED	$ED	$05
G–2	1607	71	6	$0647	$47	$06
G#–2	1703	167	6	$06A7	$A7	$06
A–2	1804	12	7	$070C	$0C	$07
B–2	1911	119	7	$0777	$77	$07
H–2	2025	233	7	$07E9	$E9	$07
C–3	2145	97	8	$0861	$61	$08
C#–3	2273	225	8	$08E1	$E1	$08
D–3	2408	104	9	$0968	$68	$09
D#–3	2551	247	9	$09F7	$F7	$09
E–3	2703	143	10	$0A8F	$8F	$0A
F–3	2864	48	11	$0B30	$30	$0B
F#–3	3034	218	11	$0BDA	$DA	$0B
G–3	3215	143	12	$0C8F	$8F	$0C
G#–3	3406	78	13	$0D4E	$4E	$0D
A–3	3608	24	14	$0E18	$18	$0E

Hangjegy	Decimális	Alsó byte	Felső byte	Hexadecimális	Alsó byte	Felső byte
B–3	3823	239	14	$0EEF	$EF	$0E
H–3	4050	210	15	$0FD2	$D2	$0F
C–4	4291	195	16	$10C3	$C3	$10
C#–4	4547	195	17	$11C3	$C3	$11
D–4	4817	209	18	$12D1	$D1	$12
D#–4	5103	239	19	$13EF	$EF	$13
E–4	5407	31	21	$151F	$1F	$15
F–4	5728	96	22	$1660	$60	$16
F#–4	6069	181	23	$17B5	$B5	$17
G–4	6430	30	25	$191E	$1E	$19
G#–4	6812	156	26	$1A9C	$9C	$1A
A–4	7217	49	28	$1C31	$31	$1C
B–4	7647	223	29	$1DDF	$DF	$1D
H–4	8101	165	31	$1FA5	$A5	$1F
C–5	8583	135	33	$2187	$87	$21
C#–5	9094	134	35	$2386	$86	$23
D–5	9634	162	37	$25A2	$A2	$25
D#–5	10207	223	39	$27DF	$DF	$27
E–5	10814	62	42	$2A3E	$3E	$2A
F–5	11457	193	44	$2CC1	$C1	$2C
F#–5	12139	107	47	$2F6B	$6B	$2F
G–5	12860	60	50	$323C	$3C	$32
G#–5	13625	57	53	$3539	$39	$35
A–5	14435	99	56	$3863	$63	$38
B–5	15294	190	59	$3BBE	$BE	$3B
H–5	16203	75	63	$3F4B	$4B	$3F
C–6	17167	15	67	$430F	$0F	$43
C#–6	18188	12	71	$470C	$0C	$47
D–6	19269	69	75	$4B45	$45	$4B
D#–6	20415	191	79	$4FBF	$BF	$4F
E–6	21629	125	84	$547D	$7D	$54
F–6	22915	131	89	$5983	$83	$59
F#–6	24278	214	94	$5ED6	$D6	$5E
G–6	25721	121	100	$6479	$79	$64
G#–6	27251	115	106	$6A73	$73	$6A

Hangjegy	Decimális	Alsó byte	Felső byte	Hexadecimális	Alsó byte	Felső byte
A–6	28871	199	112	$70C7	$C7	$70
B–6	30588	124	119	$777C	$7C	$77
H–6	32407	151	126	$7E97	$97	$7E
C–7	34334	30	134	$861E	$1E	$86
C#–7	36376	24	142	$8E18	$18	$8E
D–7	38539	139	150	$968B	$8B	$96
D#–7	40830	126	159	$9F7E	$7E	$9F
E–7	43258	250	168	$A8FA	$FA	$A8
F–7	45830	6	179	$B306	$06	$B3
F#–7	48556	172	189	$BDAC	$AC	$BD
G–7	51443	243	200	$C8F3	$F3	$C8
G#–7	54502	230	212	$D4E6	$E6	$D4
A–7	57743	143	225	$E18F	$8F	$E1
B–7	61176	248	238	$EEF8	$F8	$EE
H–7	64814	46	253	$FD2E	$2E	$FD

4.3. A táblázat használata

A hangfrekvencia-táblázat a zenében használatos hangok frekvenciáit tartalmazza. *Felépítése a következő:* Az első oszlopban vannak maguk a zenei hangok C-től H-ig. A zeneművek fordítása során soha ne feledkezzünk meg az ún. „enharmonikus" átfedésekről. Ez egy zenei szakkifejezés, ami azt jelenti, hogy temperált hangzás esetén a Cisz és Desz, a Disz és Esz, a Fisz és Gesz, a Gisz és Asz, valamint az Aisz és B megegyeznek egymással. Ha tehát az Esz hang frekvenciáját keressük, nézzük meg a Disz hangnál. A következő táblázat az enharmonikus átfedéseket sorolja fel teljes részletességgel:

Félhang	Megfelelője
Hisz, C, Deszesz	C
Cisz, Desz	C#
Ciszisz, D, Eszesz	D
Esz, Disz	D#
Diszisz, E, Fesz	E
Geszesz, F, Eisz	F
Fisz, Gesz	F#
Fiszisz, G, Aszesz	G

Gisz, Asz	G#
Giszisz, A, Heszesz	A
Aisz, B	A#
Aiszisz, H, Cesz	H

Egy fontos megjegyzés: az angol nyelvterületen a "H" helyett a "B" jelölést használják. Néhány zeneprogram eszerint íródott. Ha nem tudjuk pontosan, melyik típussal állunk szemben, a következőket kell figyelembe venni: ha a kottában előfordul a "Bb" írásmód, akkor a B jelölés egyértelműen H-t jelent.
Most térjünk vissza a hangfrekvencia-táblázathoz. A hang jele után kötőjellel elválasztott szám a hang oktávja. Értelemszerűen a 0 a legmélyebb, a 7 a legmagasabb oktávot jelöli. A 0 oktáv félhangjai egészen mélyek, míg a 7 oktáv félhangjai egészen magasak. A zenében általában az 1–6-os oktávokat használják.

A következő oszlop a frekvencia decimális értékét tartalmazza. A BASIC zenei programok mindkét beviteli forma esetén ezeket az értékeket használják. Az első forma a teljes frekvencia, amit megfelelő módon később át kell számítani. Az „alsó byte" és a „felső byte" oszlopok tartalma viszont POKE utasítással közvetlenül bevihető. Ugyanezek hexadecimális megfelelői az assembler nyelven írt programokban hasznosíthatók.

Ha pl. a C–4 frekvenciáját akarjuk beállítani, akkor a következő utasításokat használjuk:

POKE ELSŐ, 195 : POKE ELSŐ+1,16

A táblázat bármely más értékét megadhatjuk ily módon. Általánosan az alábbi utasításpárt kell használni.

POKE ELSŐ, (alsó) : POKE ELSŐ +1, (felső)

ahol az alsó és a felső a táblázat megfelelő értékei.
Hangot azonban még mindig nem hallunk! Ehhez még az alábbi utasítások szükségesek.

POKE ELSŐ + 6,240 : NYOMÁS = ELSŐ +4 : POKE NYOMÁS, 33.

Ezek hatására egy viszonylag hangos egyenletes hangot hallunk. A POKE NYOMÁS, 32 utasítás segítségével ki lehet kapcsolni a hangot.
A POKE NYOMÁS, 33 utasítást úgy képzelhetjük el, mint egy zongorabillentyű leütését, míg a POKE NYOMÁS, 32 a billentyű elengedését szimulálja. Próbáljuk ki a fenti utasításokat úgy is, hogy közben változtatjuk a hang frekvenciáját.

4.4. Egy rövid BASIC program

A fenti eljárás során zavaróan hat, hogy az utasítások újra meg újra eltűnnek a képernyőről. Ennek az a magyarázata, hogy a parancs üzemmódban bevitt POKE NYOMÁS, 33 utasítást elvégezve a gép azonnal el is felejti.
Amíg az utasítások látszódnak a képernyőn, kurzorvezérléssel újra végrehajthatjuk őket, ám van olyan eset, hogy a görgetés (scrolling) következtében egyes sorok eltűnnek a képernyő tetejéről. Ezek elkerülésére ajánljuk az alábbi program bevitelét.

```
0 REM **** P12. ****
1 :
2 :
3 HULLAM=32
10 ELSO=54272
20 HANGERO=ELSO+24
30 NYOMAS=ELSO+4
40 POKE HANGERO,15
50 INPUT F
60 F2=F/256
70 F1=F-INT(F2)*256
80 POKE ELSO,F1
90 POKE ELSO+1,F2
100 POKE ELSO+6,240
110 IF PEEK(197)=64 THEN 140
120 POKE NYOMAS,HULLAM+1
130 GOTO 110
140 POKE NYOMAS,HULLAM
150 GOTO 110
READY.
```

Ha elindítjuk a programot, egy kérdőjel jelenik meg, jelezve, hogy input adatra vár a gép. Ez az adat egy tetszőleges hangfrekvencia. A táblázat decimális oszlopából kiválaszthatunk egy értéket. A program az alsó és felső paramétereket önállóan kiszámítja. Ezután egy billentyű lenyomására megszólal az adott frekvenciájú hang, elengedésével pedig elhallgat. A programból kilépni a RUN/STOP billentyűvel lehet.

4.5. A kapu (gate) jel

Módosítsuk az előbbi BASIC program 3. sorát POKE HULLÁM, 16-ra és indítsuk újra a programot. Ekkor egy tompább hangot hallunk. A POKE HULLÁM, 128 hatására egy zaj lesz hallható. Ezeket a hullámformákat tudja a hang chip előállítani.
Nézzük meg alaposabban a POKE H NYOMÁS, HULLÁM + 1 és a POKE H NYOMÁS, HULLÁM utasításokat. Módosítsuk úgy a programunkat, hogy a 120-as sorból elhagyjuk a + 1-et, és hozzáfűzzük a 140-es sorhoz. Újraindítva a progra-

mot azt tapasztaljuk, hogy pontosan fordítva működik, mint az előző. Egy billentyű lenyomásával elhallgat a hang, felengedésével megszólal.

Mi történt a H NYOMÁS nevű tárolóval?
Nyilvánvalóan ez a hang chip irodaház olyan szobája, ahol a különböző alkalmazottak más és más feladatokat látnak el. Az egyik alkalmazottat közvetlenül +1-gyel lehet vezérelni. Ez az ún. gate vagy kapu jel.

Minden alkalommal, amikor ennek az alkalmazottnak 1 utasítást adunk, ez azt jelenti, hogy játssza le a megadott hangot (a billentyű leütése). A 0 utasítás esetén a hang elhallgat (a billentyű elengedése).

Tehát a kapu jelet minden lejátszott hang esetén be kell állítani, és egy általunk meghatározott idő után törölni kell.
A H NYOMÁS változó a hang csengését is vezérli. Az alábbiakban a hangszíneket értelmező táblázatot mutatjuk be.

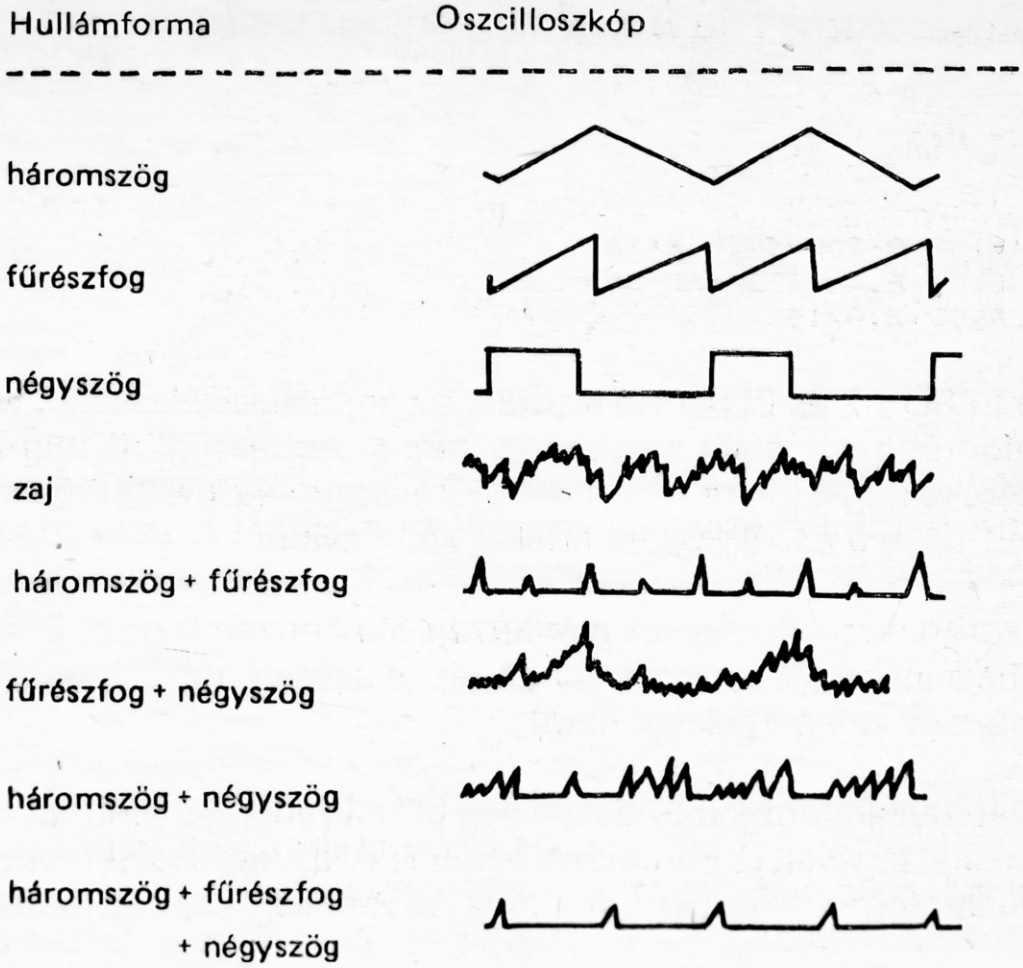
4.6. A hang chip hullámformái

A mintaprogramban szereplő HULLÁM változó megengedett értékeit az alábbi táblázat szemlélteti:

HULLÁM változó	*Hullámforma*
16	Háromszög
32	Fűrészfog
64	Négyszög
128	Zaj
16 OR 32	Háromszög + fűrészfog
32 OR 64	Fürészfog + négyszög
16 OR 64	Háromszög + négyszög
16 OR 32 OR 64	Háromszög + fűrészfog + négyszög

A hullámformák elnevezései a fizikából származnak (háromszög, négyszög, fűrészfog) és az oszcilloszkópon megjelenő képükből erednek. A fentieket az 51. oldalon lévő ábra szemlélteti.

Minden hullámforma a szinusz-függvények segítségével jól meghatározható. A legmélyebb szinuszos hangot alaphangnak nevezzük. A hang magasságát a frekvenciája határozza meg. Az alaphanghoz kapcsolódik a felharmonikusok keveréke, amelyek a hangszint határozzák meg.
Fűrészfog rezgésnél az n-edik felharmonikus 1/n-szer olyan hangos, mint az alaphang. Tehát minden felharmonikus képviselteti magát, mégpedig



$1:\frac{1}{2}:\frac{1}{3}:\frac{1}{4}:\frac{1}{5}$.... hangerővel az alaphang erősségéhez képest. A fűrészfog rezgés gazdag felharmonikus tartalma miatt „élesen", „trombitaszerűen", „csillogóan", „sugárzón" hangzik, hogy csak a gyakrabban előforduló jelzőket említsük.

Azonban ne feledjük, hogy a Commodore 64-es által generált rezgések a természetben, a valóságban nem így keletkeznek. A valódi hangok felépítése nem ilyen tiszta. Ez csak egy elektronikusan előállított rezgés, ami a hang chip többi hullámformájára is érvényes.

A háromszög rezgés jellemzője, hogy kevesebb felharmonikust tartalmaz, emiatt tompábban, furulyaszerűen cseng.

A négyszög rezgésben, vagy más néven *impulzushullám*ban csak a páratlan számú felharmonikusok szerepelnek. Ezért egy öblös oboaszerű hangzást hallunk. Az egyes felharmonikusok hangereje az impulzusszélesség változtatásával módosítható.

Érdekes hangzást érhetünk el az impulzusszélesség folyamatos változtatásával. Ezt a folyamatot PWM-nek (Pulse Width Modulation), vagy magyarul impulzusszélesség modulációnak nevezzük. Ezt az előző mintaprogram segítségével úgy állíthatjuk elő, hogy a 3-as sort POKE HULLÁM, 64-re módosítjuk és begépeljük

az alábbi sorokat (tkp. a P.13-as program 115-ös, 116-os és 117-es soráról van szó):

```
0 REM **** P13. ****
1 :
2 :
115 A=A+16:IF A=256 THEN A=16
116 POKE ELSO+2,(A*16) AND 255
117 POKE ELSO+3,A/16
READY.
```

A hang chip ELSŐ + 2 és ELSŐ + 3 szobája az impulzusszélességet vezérli. Négyszög hullámforma használata esetén ezeknek a szobáknak mindig kell kapniuk valamilyen értéket, egyébként nem hallható a hang. Az ELSŐ + 3 regiszter felső 4 bitje 0, ezért csak 0 és 15 közötti értéket kell bevinni.

Az eddigiek során nem értelmeztük a *fehérzajt*, mint hullámformát. Ennek jellemzője, hogy benne minden felharmonikus hangereje azonos, tehát frekvenciatartománya folytonos, ami zajként jelenik meg.

Az említett alap hullámformákból elméletileg bármilyen hullámformát képezhetünk a hang chip segítségével. Ezt a chipen belül az AND kapcsolat oldja meg. A legtöbb ilyen hullámforma azonban igen halk és zeneileg csak korlátozottan alkalmazható.

4.7. Az ADSR programozása

A mintaprogramunkban egy billentyű lenyomása után úgy érezzük, hogy a hang azonnal megszólal, a billentyű felengedésével pedig elhallgat. A valóságban ezek a folyamatok persze némi időt igényelnek. Ezeket a bekapcsolási, illetve kikapcsolási időket lehet a hang chip segítségével vezérelni.
Mintaprogramunk 100-as sorában láthatjuk, hogy eddig a teljes időt kihasználtuk. Ez az irodaház azon szobája, ahol ezeket az időket vezérlik. Programunkba szúrjuk be a 91–95-ös sorokat és módosítsuk a 100-as sort az alábbiak szerint:

```
0 REM **** P14. ****
1 :
2 :
91 A=0
92 D=0
93 S=15
94 R=0
95 POKE ELSO+5,A*16+D
100 POKE ELSO+6,S*16+R
READY.
```

Programunkat újraindítva nem tapasztalunk semmi különbséget. Ez azért van, mert a felfutási és lecsengési időket a legkisebbre állítottuk. Azért írhattuk eddig minden további nélkül, hogy POKE ELSŐ + 6.240, mert 15*16 = 240.

A paraméterek gyors változtatása érdekében módosítsuk a 91–94-es sorokat az alábbiak szerint.

```
0 REM **** P15. ****
1 :
2 :
91 INPUT A
92 INPUT D
93 INPUT S
94 INPUT R
READY.
```

A program elindítása után most értelemszerűen négy számot kell megadni a hangfrekvencia után a megfelelő sorrendben. A számoknak 0 és 15 közé kell esniük. Ha ez a megoldás kényelmetlen, változtassuk meg az 50-es sort a következőképpen:

```
50 F=4291
```

Ezzel pl. a C4 hangot állítottuk be.

A fentiekből láthatjuk, hogy az INPUT utasítás használatával minden programfutás esetén új értékeket adhatunk meg. Ha viszont rögzített paraméterekkel akarjuk a programot futtatni, akkor a kívánt értéket egyenlőségjellel kell beállítani.

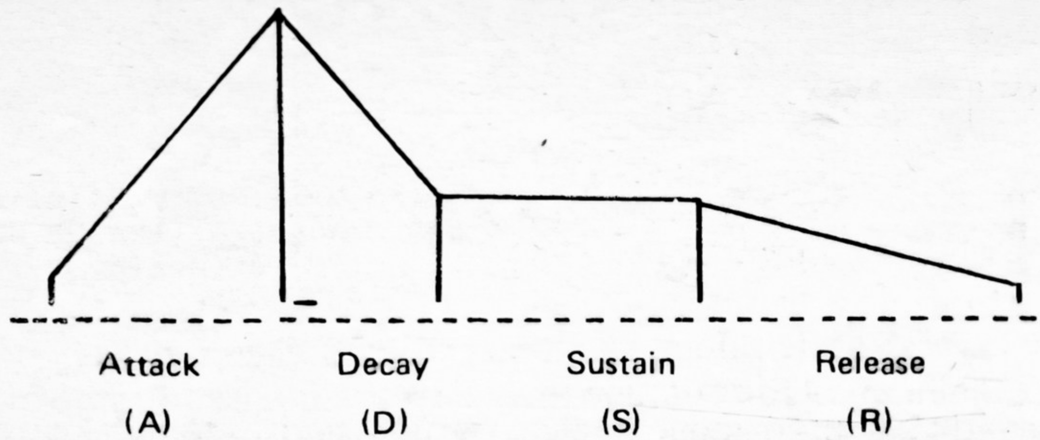
A négy paramétert nem ok nélkül jelöltük A, D, S, R betűkkel, a szintetizátoroknál már foglalkoztunk velük (ADSR érték). Ezek a paraméterek határozzák meg egy hang időbeni lefutását.

Mivel e téma egzakt leírása még túl nehéz, helyette kísérletezzünk az INPUT utasításokkal bővített mintaprogramunkkal. Változtassuk az A, D, S, R értékeket, a 3-as sorba pedig a HULLÁM értékének írjunk 16-ot, 32-t vagy 64-et.

INPUT				HANGZÁS
A	D	S	R	
1	9	0	0	Rövid, zongoraszerű hang
5	10	5	10	Hosszabb, hegedűszerű hang
13	8	8	13	Lassan felhangzó és lecsengő
0	5	0	0	Rendkívül rövid hang
0	0	0	0	Nincs hang

A második és a harmadik beállítás kiválóan szemlélteti a hangfelfutás egyes fázisait. Egy billentyű lenyomásával a hang lassan felfut (Attack), elér egy max. értéket és egy jóval halkabb értékre lecseng (Decay), majd egy meghatározott hangerőn marad mindaddig (Sustain), amíg a billentyűt el nem engedjük (Release). Ezután a hang végleg lecseng. A hangerő-görbe a következőképpen szemléltethető:

Az ADSR burkológörbe



A fenti görbét *burkológörbé*nek nevezzük, melynek alakját az ADSR érték határozza meg. Az értékek módosításával számtalan különböző burkológörbe állítható elő.
Próbáljunk ki minél több kombinációt!
Hallható lesz, hogy mindegyik tud valami újat nyújtani.
Pontosan ebben rejlik a Commodore 64-es hang chipjének nagy erőssége.
A szintetizátoron beállított értékek kombinációjával mindig lehet valami újat fölfedezni.
Némely fenti érték jól ismert hangzást eredményez. Ha pl. kis A értéket választunk, a hang igen gyorsan felfut és így szimulálja a spinét vagy a csemballó hangját.

4.8. Az ADSR értékek pontos jelentése

Ismerkedjünk meg az ADSR értékek pontos jelentésével. Az ADSR az Attack, Decay, Sustain és Release angol kifejezések rövidítése. A fogalmak a hanglefutás egyes fázisait jelölik, melyeket a hozzátartozó érték jellemez.
Az egyes fogalmak részletes jelentése a következő:

1. ATTACK: A hang felfutási idejét jellemző érték

Ez az angol szó rázendítést jelent. Értéke meghatározza, hogy egy hang mennyi idő alatt éri el maximális hangerejét.
A „természetes" hangszereknél előforduló attack értékek néhány milliszekundumtól pár másodpercig terjedhetnek.

Rövid attack értékkel rendelkeznek a húros hangszerek, ahol a húrokat pengetik vagy ütik. Ilyenek a hárfa, a gitár és a legtöbb billentyűs hangszer.

A fúvós hangszerek hosszabb attack idővel rendelkeznek. Ez azért van, mert a zenésznek levegőt kell fújnia a hangszerbe, mielőtt a hang megszólal.

A vonós hangszerek rázendítési ideje még hosszabb. Ennek oka a húrok megszólaltatásának módjában keresendő. A zenész a vonóval jóval hosszabb idő alatt éri el a hangerő maximumát az előző hangszerekhez képest.

A leghosszabb rázendítési idejű zajok között találjuk a szél hangját generáló zajokat. Ezek hangereje igen lassan növekszik, ill. csökken. Minél rövidebb idejű az attack, annál pregnásabb az ún. „peak" (csúcs) a hang elején, ami leginkább a billentyűs hangszereket jellemzi.

Rendkívül gyors (néhány milliszekundum) attack idejű hangokat csak elektromos úton lehet előállítani. Jó példa erre az elektromos orgona, amelynél egy billentyű lenyomása után azonnal felcsendül a hang.

2. Decay: a hang hanyatlását meghatározó érték.

Ez az angol szó hanyatlást, visszaesést jelent. Értéke azt határozza meg, hogy egy hang az attack által elért maximális értéktől milyen gyorsan csökken le egy meghatározott értékre. A decay fázist mintegy az attack tükörképének lehet tekinteni.
Mindkét fázis a hangerő felfutását befolyásolja először a 0 szintről a maximumig, majd onnan egy alacsonyabb szintig a sustain értékéig. A decay értéke rendszerint hosszabb mint az attacké, mivel a már gerjesztett hang lassabban fog lecsengeni, mint amennyi idő alatt felfutott.

3. Sustain: a hang kitartási szintjét határozza meg.

A sustain (= angol: kitart) értéke az attack és decay fázisok után határozza meg a hang értékét. Gyakran beszélünk sustain szintről is. Fontos, hogy a sustain értékét mindig a maximális hangerőhöz képest vizsgáljuk. A sustain értéke ugyanis nem abszolút hangerősség, hanem egy viszonyszám.

Egy hang lefutásának ADSR értékekkel való leírásánál az attack és a decay meghatározása mellett, melyek az adott fázisok időintervallumait adják meg, definiálni kell még a sustain és a maximális hangerő értékét is. Ez utóbbi az attack fázis végén a decay elején jön létre. A sustain érték megadja, hogy a decay végén a hangerő hányad része a maximálisnak.
Így a 0 sustain érték azt jelenti, hogy a decay fázis után semmilyen hang nem hallatszik. A 8-as érték bevitele esetén a hang félig lehalkul. A 15-ös érték azt jelenti, hogy az attack után a hangerősség a maximális értéken marad.

A sustain értékének megfelelő hangerőn tartás idejét nem az ADSR vezérli. Erre van a kapu jel. Mindaddig, amíg a kapu = 1, a hang a sustain szinten marad. Ha a kapu 0-ra vált át, a hang átmegy a release (lecsengési) fázisba.

A billentyűs hangszerek esetében a sustain értéke általában 0. Ennek az az oka, hogy hiába tartjuk lenyomva egy hang billentyűjét, az nem tartja a hangot. A hang megszólal (attack), majd lecseng (decay) és vége.
A billentyűs hangszerek között kivételt képez az elektromos orgona. Ennél az

összes ADSR érték 0, a sustain kivételével, ami 15, vagyis maximális. Ennél a hangszernél ugyanis nincs attack és decay fázis. Ha a kapu = 1, akkor a hang azonnal eléri a sustain szintet, ami egyben a maximális hangerőt jelenti. A kapu = 0 esetén a hang azonnal elhallgat. Így a hangerő nem lépcsőzetesen változik, csak a teljes hangerőt lehet a pedál segítségével módosítani.

4. Release: a hang lecsengési idejének értéke

A release (= angol: elereszt, elenged) azt az időintervallumot határozza meg, amíg a hangerő a sustain szintjéről 0-ra csökken. A mintaprogram viszonylag nagy release értéket tartalmaz, így a billentyű elengedése után a hang még sokáig szól. A természetes hangszereknél a release érték általában rövid. Kevés olyan hangszer van, amely a hang létrehozása után még tovább cseng.

4.9. Attack–Decay–Release időtáblázat

Az alábbi táblázat a hang chip segítségével elérhető rázendítési, hanyatlási és lecsengési időket tartalmazza. Az „ms" rövidítés milliszekundumokat (ezredmásodperceket), az „s" pedig másodperceket jelent.

Érték	Attack	Decay	Release
0	2 ms	6 ms	6 ms
1	8 ms	24 ms	24 ms
2	16 ms	48 ms	48 ms
3	24 ms	72 ms	72 ms
4	38 ms	114 ms	114 ms
5	56 ms	168 ms	168 ms
6	68 ms	204 ms	204 ms
7	80 ms	240 ms	240 ms
8	100 ms	300 ms	300 ms
9	250 ms	750 ms	750 ms
10	500 ms	1,5 s	1,5 s
11	800 ms	2,4 s	2,4 s
12	1 s	3,0 s	3,0 s
13	3 s	9,0 s	9,0 s
14	5 s	15,0 s	15,0 s
15	8 s	24,0 s	24,0 s

4.10. A hang chip oszcillátorainak összefoglalása

Az eddigiek során egyszerre mindig csak egy hangot játszottunk le, pedig a hang chip három hangot képes egy időben előállítani. Minden hanghoz ugyanannyi, 52 alkalmazott tartozik. Ezek 7 szobában helyezkednek el, melyek szobacsoportot képeznek. Ezekből összesen három van.

Egy-egy ilyen szobacsoportot más néven oszcillátornak nevezünk. Az oszcillátor szó tehát mindig egy ilyen 7 szobából álló csoportot jelent. Ezek a szobacsoportok az irodaházban egymás között helyezkednek el. Így a 29 szobából már 7*3 = 21 szobát ismerünk.

Az alábbi táblázat ezeket foglalja össze:

Sorszám	1. oszcillátor
0	Frekvencia alsó byte
1	Frekvencia felső byte
2	Impulzusszélesség alsó byte
3	Impulzusszélesség felső byte
4	Hullámforma vezérlőbitek
5	Attack érték, Decay érték
6	Sustain érték, Release érték

Sorszám	2. oszcillátor
7	Frekvencia alsó byte
8	Frekvencia felső byte
9	Impulzusszélesség alsó byte
10	Impulzusszélesség felső byte
11	Hullámforma vezérlőbitek
12	Attack érték, Decay érték
13	Sustain érték, Release érték

Sorszám	3. oszcillátor
14	Frekvencia alsó byte
15	Frekvencia felső byte
16	Impulzusszélesség alsó byte
17	Impulzusszélesség felső byte
18	Hullámforma vezérlőbitek
19	Attack érték, Decay érték
20	Sustain érték, Release érték

A fentiekből láthatjuk, hogy az oszcillátorok egymással teljesen egyenértékűek. Az 1. oszcillátor a 0–6. sorszámú, a 2. oszcillátor a 7–13. sorszámú, a 3. oszcillátor pedig a 14–20. sorszámú szobákat foglalja le.

A táblázat tartalmaz egy új kifejezést, a *vezérlőbit* fogalmát. Ez a kifejezés a számítógépes terminológiából való, jelentését már többé-kevésbé ismerjük. Eddig egyszerűen csak a szót használtuk. Egy ilyen külön ismertetett bit volt a kapu jel. A többi vezérlőbitet később tárgyaljuk.

Ha már a szakterminológiánál tartunk, egységesítsük a fogalmakat. Az irodaház szobái helyett használjuk a „vezérlő regiszter" vagy csak egyszerűen „regiszter" szavakat. Amikor a szobákat a megfelelő alkalmazottakkal azonosítottuk, akkor tulajdonképpen a regiszterekre és a vezérlőbitekre gondoltunk. A továbbiakban mindkét megnevezési formát fogjuk alkalmazni.

4.11. A hang chip szűrőjének működési elve és beállítása

A három oszcillátor mellett a hang chip még egy szűrővel is rendelkezik. A szűrő működésének elve egyszerű, melyet jól szemléltet a sztereó berendezések hangszínszabályzója. Ezen szabályzók egyike képes kiemelni, vagy elnyomni a basszust, míg a másik a magas hangokkal tudja ugyanezt. Ezek a szűrők mindig csak egy meghatárotott frekvenciát tudnak kiemelni vagy elnyomni. A hang chip szűrőjével azonban konkrét frekvenciákat tudunk módosítani.

A hang chip által előállított hullámformák esetében gyakoriak a felharmonikusok. Ezért annyira tiszta és csillogó a hangzásuk, hogy szinte már természetellenes. A hangzásnak a természetes hangszerekhez való közelítése érdekében lehetőség nyílik a hangzás lágyabbá tételére a felhangok „levágásával". Ennek módszere, hogy az oszcillátorból jövő hullámokról a szűrő átengedi a mélyhangokat és levágja a magasakat. Ezért ezt a szűrőt *aluláteresztő*nek is nevezik.

A szűrő beállítható *felüláteresztő*nek is, mely átengedi a magas hangokat, de a mélyeket levágja. Mindkét szűrőtípusnál beállíthatjuk azt a frekvenciát, melytől kezdve a szűrő a felharmonikusokat nem engedi át, levágja.
A harmadik szűrőtípussal, melyre a hang chip szintén beállítható, egy meghatározott ún. frekvenciasáv ereszthető át, vagyis mind a mély, mind a magas frekvencia levágható. A sávszűrő a leggyakrabban előforduló frekvenciát engedi át.

A szűrési frekvencián és a szűrőtípuson kívül beállítható még a szűrő rezonanciafrekvenciája is, melyhez a szűrőt a hang chip negyedik oszcillátoraként kell elképzelni. Így a szűrőn ugyanúgy be lehet állítani egy frekvenciát, mint a többi három oszcillátoron. A rezonancia értéke meghatározza, hogy a szűrő milyen mértékben működik oszcillátorként. A rezonancia nulla érték esetében csak a frekvenciát vágja le, értékének növelésével azonban a szűrési frekvencia amplitúdója növekszik.

A szűrő-rezonancia maximális értéke 15. A szűrőbe vezetett oszcillátorok hangzása a szűrési frekvencia következtében módosul.

Az elmondottakból érezhető, hogy a szűrő alkalmazása nagymérvű hangzásváltozást idézhet elő, mivel az oszcillátorok hullámformáját sokféle módon tudjuk módosítani.

A következő táblázat azt mutatja, hogy a hang chip mely regiszterei befolyásolják a szűrő működését. Ezekben a regiszterekben vannak az előzőekben leírt funkciókat vezérlő bitek.

	7. bit	6. bit	5. bit	4. bit	3. bit	2. bit	1. bit	0. bit
21						2. frekv.	1. frekv.	0. frekv.
22	10. frekv.	9. frekv.	8. frekv.	7. frekv.	6. frekv.	5. frekv.	4. frekv.	3. frekv.
23	3. rez.	2. rez.	1. rez.	0. rez.	FILTEX	3. szűrő	2. szűrő	1. szűrő
24	3. KI	felülát.	sávát.	alulát.	3. hangerő	2. hangerő	1. hangerő	0. hangerő

Az előbbi táblázat formájában vázolt hang chip iroda „szobái" (regiszterek) olyan vezérlőbiteket tartalmaznak, melyek meghatározzák a szűrő rendeltetésszerű működését. A szűrőfrekvenciákat beállító bitek a 21. és a 22. regiszterben vannak. A szűrőfrekvencia szélessége miatt (csak 0 és 2047 közötti értékeket lehet beállítani) a 21. regiszterben nincs minden bit definiálva.

A kívánt szűrőtípust a 24. regiszter 6., 5. és 4. bitjével lehet kiválasztani. A bitek külön-külön beállíthatók és törölhetők. Ezzel lehetőség nyílik kombinált szűrőtípusok létrehozására az alábbi módon.

6. bit	5. bit	4. bit	Szűrőtípus
0	0	0	Lezáró szűrő
0	0	1	Aluláteresztő
0	1	0	Sávszűrő
0	1	1	Aluláteresztő és sávszűrő
1	0	0	Felüláteresztő
1	0	1	Aluláteresztő és felüláteresztő
1	1	0	Felüláteresztő és sávszűrő
1	1	1	Mindent áteresztő

A *lezáró* szűrőtípus, (a bitek értéke nulla) beállításával azon oszcillátor frekvenciája, melyet ezen a szűrőn vezetünk át, nem hallható.

A *mindent áteresztő* szűrőtípus (a bitek értéke 1) minden frekvenciát átereszt szűrés nélkül. Hasonlóan az előbbi szűrőtípushoz a szűrőfrekvenciának itt sincs jelentősége.

A 23. regiszterben található 4. rez. bittel a szűrőrezonancia állítható be 0 és 15 közötti értékre. A regiszter többi bitje azt határozza meg, hogy melyik oszcillátor frekvenciája kerül átvezetésre a szűrőn.

A „FILTEX" beállításával lehetőség nyílik arra, hogy „audio-in" jelet szűrjünk. Ehhez azonban a hang chip hardver oldalának mélyebb ismerete szükséges.

A 24. regiszter tartalmaz még néhány ún. hangerő-vezérlő bitet, melyeknek semmi közük a szűrő beállításához. A "3.KI" bittel az ötödik fejezetben találkozunk ismét.

4.12. A szinkronizáció és a gyűrűsmoduláció (ringmoduláció)

A hang chip szűrőjének alkalmazása lehetővé tette, hogy az oszcillátorok célirányos felhasználását igényünknek megfelelően változtathassuk a felharmonikusok eltávolításával. Ezeken túlmenően további két másik lehetőség is van az oszcillátorokból jövő jelek megváltoztatására, mégpedig a szinkronizáció és a gyűrűsmoduláció.

A szűrés során csak egy oszcillátor jelét módosítottuk. A szinkronizálással és a gyűrűsmodulációval azonban tudjuk egy másik oszcillátor jelétől függően az egyik, vagy mindkét másik oszcillátor kimenő jelét változtatni. Ez úgy történik, hogy a hangforrásul szolgáló oszcillátor jelét egy másik oszcillátor jele befolyásolja. Azt az oszcillátort, melynek a jelét módosítjuk, bázisoszcillátornak, amellyel módosítunk, vezérlő oszcillátornak nevezzük.

A szinkronizáció alapja, hogy az előbb tárgyalt mindkét oszcillátor eltérő frekvenciájú és hullámformájú jeleket bocsát ki. Ezek a hullámok egészen eltérő alakkal és formával rendelkezhetnek.

Ha a bázisoszcillátort a vezérlő oszcillátor szinkronizálja, akkor az utóbbi egy start/stop kapcsolóként funkcionál. Vagyis ha a vezérlő oszcillátor jelének egy periódusa lefutott és új hullám kezdődik, akkor a bázisoszcillátornak egy start jelet továbbít, ami biztosítja az oszcillátorok együttes indulását.

Abban az esetben, ha az oszcillátorok hullámformája és frekvenciája azonos, akkor az együttes futás teljes mértékben biztosított. Ellenkező esetben, mely a hullámforma vagy a frekvencia eltérését jelenti, a start jel a bázisoszcillátort egy úl hullám megkezdésére kényszeríti. Ennek hatására a bázisoszcillátor a rezgései-

nek egy részét fejezi be, mert a két oszcillátor hullámainak eltérése miatt mindig elölről indul.

Ez az eljárás különlegességnek számító hullámformákat is létrehozhat. Gyakorlatilag korlátlan lehetőség előtt állunk, mert a vezérlőregiszter minden frekvenciája és hullámformája más-más hatást vált ki.

A gyűrűsmoduláció esetében a hang chipen belül a bázis- és a vezérlő oszcillátor rezgésének számértéke digitálisan összeszorzódik és a bázisoszcillátor kimenetére kerül.

Szinte elképzelhetetlen hullámforma keletkezik akkor, ha a két oszcillátor frekvenciájának értéke közel áll egymáshoz. Ennek eredménye gyakran a fémes és harangszerű hangzás, mert sok nem harmonikus felhangot tartalmaz.

Az alábbi vezérlőbitek hozzák létre az előzőekben leírtakat:

3. bit	2. bit	1. bit	0. bit
Teszt-bit	Gyűrűsmoduláció	Szinkronmoduláció	kapu

Ez a négy bit mindhárom oszcillátornál megtalálható. A vezérlőbiteket oszcillátoronként külön-külön is be lehet állítani, ill. törölni. Ezért mind a szinkronizálásra, mind a gyűrűsmodulációra nyolc lehetőség áll rendelkezésre, melyeket a következő táblázat tartalmazza.

1. oszc.	2. oszc.	3. oszc.	Eredmény
0	0	0	Nincs szinkronizáció
1	0	0	Az 1. oszcillátort a 3. szinkronizálja
0	1	0	A 2. oszcillátort az 1. szinkronizálja
0	0	0	A 3. oszcillátort a 2. szinkronizálja
1	1	0	Az 1. és 2. oszcillátort a 3. szinkronizálja
0	1	1	A 2. és 3. oszcillátort az 1. szinkronizálja
1	0	1	Az 1. és 3. oszcillátort a 2. szinkronizálja
1	1	1	Mindhárom oszcillátor szinkronban fut

1. oszc.	2. oszc.	3. oszc.	Eredmény
0	0	0	Nincs gyűrűs moduláció
1	0	0	Az 1. oszcillátort a 3. modulálja
0	1	0	A 2. oszcillátort az 1. modulálja
0	0	1	A 3. oszcillátort a 2. modulálja
1	1	0	Az 1. és 2. oszcillátort a 3. modulálja
0	1	1	A 2. és 3. oszcillátort az 1. modulálja
1	0	1	Az 1. és 3. oszcillátort a 2. modulálja
1	1	1	Minden oszcillátor mindegyiket modulálja

A táblázatban lévő teszt-bit értékét a felhasználó állítja be oszcillátoronként. Ha a teszt = 0, akkor minden normálisan működik, ha viszont 1, akkor az érintett oszcillátor kikapcsol. Ha a „zajt" egy másik hullámformával akarjuk kombinálni, akkor a hang chip a teszt-bit értékét 1-re változtatja, mivel ilyen kombinációra a hang chip nem alkalmas.
A teszt-bit újbóli nullázása után az előző állapotnak megfelelően működik minden.

A hang chip valamennyi regiszterének megismerését követően összefoglalásként még bemutatunk két programot.
Az egyik program egy gyűrűsmodulációs effektust állít elő a szűrő szemléltetése érdekében, a másik pedig egy regisztertáblázatot, mely a hang chip összes regiszterét tartalmazza.
Befejezésként pedig azok a fejezetek következnek, melyek haladók számára nyújtanak információt a regiszterek használatára vonatkozóan.

```
0 REM **** P16. ****
1 :
2 :
3 REM OSSZEFOGLALO MINTAPROGRAM
4 REM AZ ADSR-ERTEKEK ES A SZUROK
5 REM BEALLITASARA
6 HULLAM=64
7 REM HULLAMFORMA
8 ELSO=54272
9 POKE ELSO+3,8
10 REM IMPULZUSSZELESSEG
15 HANGERO=ELSO+24
20 NYOMAS=ELSO+4
40 INPUT"FREKVENCIA";F
50 F2=F/256
60 F1=F-INT(F2)*256
70 POKE ELSO+1,F2
80 INPUT"A,D,S,R";A,D,S,R
90 POKE ELSO+5,A*16+D
100 POKE ELSO+6,S*16+R
110 INPUT"SZURO KI (0) VAGY BE (1)";P
120 SZURO=15
130 IF P=0 THEN 270
140 INPUT"FELULATERESZTO (0/1)";P
150 IF P THEN SZURO=SZURO OR 64
160 INPUT"SAVATERESZTO (0/1)";P
170 IF P THEN SZURO=SZURO OR 32
180 INPUT"ALULATERESZTO (0/1)";P
190 IF P THEN SZURO=SZURO OR 16
200 INPUT"FREKVENCIA (0-2047)";FRQ
210 FH=INT(FRQ/8)
220 FL=FRQ-FH*8
230 POKE ELSO+21,FL
240 POKE ELSO+22,FH
250 INPUT"REZONANCIA (0-15)";REZ
260 REZ=REZ*16+1
270 POKE ELSO+23,REZ
280 POKE ELSO+24,SZURO
```

```
300 IF PEEK(197)=64 THEN330
310 POKE NYOMAS,HULLAM+1
320 GOTO 300
330 POKE NYOMAS,HULLAM
340 GOTO 300
READY.
```

```
0 REM **** P17. ****
1 :
2 :
5 ELSO=54272
10 FOR I=0 TO 7
20 POKE ELSO+24,15
30 POKE ELSO+5,0
40 POKE ELSO+6,240
50 POKE ELSO+4,16+4+1
60 READ A:IF A=0 THEN 500
70 A=A/2↑I
80 L=INT(A/256):H=INT(A-(256*L))
90 IF TI<TM+(8-I)*2 THEN 90
100 TM=TI
110 POKE ELSO+1,L
120 POKE ELSO+15,H
130 GOTO 50
500 RESTORE
510 NEXT
520 FOR C=1 TO 1000:NEXT
530 POKE ELSO+24,0
900 DATA 64814,61176,57743,54502,51443,48556,45830,43258,40830,38531
910 DATA 36376,34334
320 DATA 0
READY.
```

4.13. Regisztertáblázat

DEC.	HEX.	Funkció	7. bit	6. bit	5. bit	4. bit
0 54272	$D400	Frekvencia alsó	f1 7	f1 6	f1 5	f1 4
1 54273	$D401	Frekvencia felső	f1 15	f1 14	f1 13	f1 12
2 54374	$D402	Imp. szélesség alsó	p1 7	p1 6	p1 5	p1 4
3 54275	$D403	Imp. szélesség felső	....	....	....	....
4 54276	$D404	Hullámforma	zaj	imp.	fűrész	háromszög
5 54277	$D405	Attack/Decay	a1 3	a1 2	a1 1	a1 0
6 54278	$D406	Sustain/Release	s1 3	s1 2	s1 1	s1 0
7 54279	$D407	Frekvencia alsó	f2 7	f2 6	f2 5	f2 4
8 54280	$D408	Frekvencia felső	f2 15	f2 14	f2 13	f2 12
9 54281	$D409	Imp. szélesség alsó	p2 7	p2 6	p2 5	p2 4

DEC.		HEX.	Funkció	7. bit	6. bit	5. bit	4. bit
10	54282	$D40A	Imp. szélesség felső	....	....	....	....
11	54283	$D40B	Hullámforma	zaj	imp.	fűrész	háromszög
12	54284	$D40C	Attack/Decay	a2 3	a2 2	a2 1	a2 0
13	54285	$D40D	Sustain/Release	s2 3	s2 2	s2 1	s2 0
14	54286	$D40E	Frekvencia alsó	f3 7	f3 6	f3 5	f3 4
15	54287	$D40F	Frekvencia felső	f3 15	f3 14	f3 13	f3 12
16	54288	$D410	Imp. szélesség alsó	p3 7	p3 6	p3 5	p3 4
17	54289	$D411	Imp. szélesség felső	....	....	....	....
18	54290	$D412	Hullámforma	zaj	imp.	fűrész	háromszög
19	54291	$D413	Attack/Decay	a3 3	a3 2	a3 1	a3 0
20	54292	$D414	Sustain/Release	s3 3	s3 2	s3 1	s3 0
21	54293	$D415	Szűrőfrekv. alsó	....	....	....	....
22	54294	$D416	Szűrőfrekv. felső	fc 10	fc 9	fc 8	fc 7
23	54295	$D417	Szűrőrezonancia	REZ 3	REZ 2	REZ 1	REZ 0
24	54296	$D418	Hangerő és más	3 KI	Felül	Sáv.	Alul
25	54297	$D419	Potenciométer 1.	7	6	5	4
26	54298	$D41A	Potenciométer 2.	7	6	5	4
27	54299	$D41B	3. oszc. felső byte	f3 15	f3 14	f3 13	f3 12
28	54300	$D41C	3. Hf. felső byte	e3 7	e3 6	e3 5	e3 4

DEC.		HEX.	Funkció	3. bit	2. bit	1. bit	0. bit
0	54272	$D400	Frekvencia alsó	f1 3	f1 2	f1 1	f1 0
1	54273	$D401	Frekvencia felső	f1 11	f1 10	f1 9	f1 8
2	54274	$D402	Impulzus alsó	p1 3	p1 2	p1 1	p1 0
3	54275	$D403	Impulzus felső	p1 11	p1 10	p1 9	p1 8
4	54276	$D404	Hullámforma	teszt	gy. mod	szinkr.	kapu
5	54277	$D405	Attack/Decay	d1 3	d1 2	d1 1	d1 0
6	54278	$D406	Sustain/Release	r1 3	r1 2	r1 1	r1 0
7	54279	$D407	Frekvencia alsó	f2 3	f2 2	f2 1	f2 0
8	54280	$D408	Frekvencia felső	f2 11	f2 10	f2 9	f2 8
9	54281	$D409	Impulzus alsó	p2 3	p2 2	p2 1	p2 0
10	54282	$D409	Impulzus felső	p2 11	p2 10	p2 9	p2 8
11	54283	$D40B	Hullámforma	teszt	gy. mod	szinkr.	kapu

DEC.	HEX.	Funkció	3. bit	2. bit	1. bit	0. bit
12 54284	$D40C	Attack/Decay	d2 3	d2 2	d2 1	d2 0
13 54285	$D40D	Sustain/Release	r2 3	r2 2	r2 1	r2 0
14 54286	$D40E	Frekvencia alsó	f3 3	f3 2	f3 1	f3 0
15 54287	$D40F	Frekvencia felső	f3 11	f3 10	f3 9	f3 8
16 54288	$D410	Impulzus alsó	p3 3	p3 2	p3 1	p3 0
17 54289	$D411	Impulzus felső	p3 11	p3 10	p3 9	p3 8
18 54290	$D412	Hullámforma	teszt	gy. mod	szinkr.	kapu
19 54291	$D413	Attack/Decay	d3 3	d3 2	d3 1	d3 0
20 54292	$D414	Sustain/Release	r3 3	r3 2	r3 1	r3 0
21 54293	$D415	Szűrőfrekv. alsó	. . . .	fc 2	fc 1	fc 0
22 54294	$D416	Szűrőfrekv. felső	fc 6	fc 5	fc 4	fc 3
23 54295	$D417	Szűrőrezonancia	FILTEX	szűrő 3	szűrő 2	szűrő 1
24 54296	$D418	Hangerő és más	hang-erő 3	hang-erő 2	hang-erő 1	hang-erő 0
25 54297	$D419	Potenciométer 1.	3	2	1	0
26 54298	$D41A	Potenciométer 2.	3	2	1	0
27 54299	$D41B	3. oszc. felső byte	f3 11	f3 10	f3 9	f3 8
28 54300	$D41C	3. Hf. felső byte	e3 3	e3 2	e3 1	e3 0

4.14. Röviden a regisztertáblázatról

A rendelkezésre álló hely korlátozottsága miatt a táblázatot két részre bontottuk. Az első része a 7–4. biteket, a második része pedig a 3–0. biteket tartalmazza.

Az első, második és a harmadik oszcillátor esetében a 0–6., a 7–13. és a 14–20. sorszámú regiszterek funkciója azonos. A 3., 10. és a 17. számú regiszterek 3–0. bitjei az impulzus szélességét vezérlik, a 7–4. bitek pedig nincsenek definiálva. A 21. regiszter felső öt bitje nincs definiálva, az alsó három bit pedig a szűrőfrekvenciát vezérli.
A pontokkal jelölt bitek nincsenek definiálva.

A 0–24. számú regiszterekbe csak írni lehet a POKE utasítással. A PEEK utasítással nem lehet kiolvasni, eredményként mindig 0 értéket kapunk. A 25–28. regiszterekből viszont csak olvasni lehet, kizárólag a PEEK utasítással. A 24. és a 25. regiszterekkel az A/D átalakítót lehet lekérdezni.

A harmadik oszcillátor felső nyolc bitjét a 27. regiszterből lehet kiolvasni.
A 28. regiszterből a harmadik oszcillátor burkológörbéjének felső nyolc bitje olvasható ki. Ezeket a regisztereket modulációs célra is fel lehet használni. A bemutató jellegű példákat az 5. fejezet tartalmazza.

4.15. Néhány program és megjegyzés haladók számára

A processzor és az I/O elemek kapcsolatát biztosító busz működését megérteni elég nehéz. Éppen ezért nézzünk néhány programot és némi magyarázatot.
A központi memóriában előállított I/O koncepció azt a célt szolgálja, hogy az I/O cellákat memóriahelyenként lehessen címezni. Az $D000–$DFFF címtartománban négy I/O cella található. A két 6526-os, valamint a 6569-es videochip és a 6581-es hang chip hozzáférési idejét zökkenőmentesen összekombinálni rendkívül nehéz feladat. Ugyanis a négy chip mindegyikének hozzá kell férnie a számukra kijelölt memóriahelyhez a processzor működésének akadályozása nélkül. Ez mégis megoldott néhány „zökkenőtől" eltekintve. Mindezek ellenére a videochip hibája miatt a sprite vagy a karakterkészlet alkalmazásakor sokan bosszankodtak már.

A hang chip esetében az jelenti a problémát, hogy a regisztereket csak kiolvasni lehet. Tehát a PEEK utasítással nem nyerjük vissza a POKE utasítással előzőleg bevitt tartalmat. A probléma megszüntetése érdekében lehet alkalmazni a „READ–WRITE REGISTER INTERRUPT" programot, melynek listája a következő oldalon található.

A BASIC betöltő alkalmazása esetén a RUN parancs után a SYS utasítás aktivizálja a megszakítást. Nyugodtan dolgozhatunk a forrásszöveggel az assemblerben történő programozáskor. Az 5. fejezetben szereplő assembler programokra vonatkozó útmutatót vegyük figyelembe.

A példaprogram egy megszakító programból áll, mely átmásolja a $02C0–$02D8 között elhelyezett 25 WRITE ONLY regiszter tartalmát az $D400–$D418 közötti tartományba. A megszakító alkalmazása esetén a címkéket kvázi READ–WRITE regiszterként lehet használni. A BASIC mintaprogramokban az ELSŐ = 54272 helyett (54272 dec. = $D400) egyszerűsítve ELSŐ = 704 írható (704 dec. = $02C0).

```
0 REM **** P18. ****
1 :
2 :
3 OPEN4,4
4 SYS 49152
5  OPT OO,P4
6 *=$033C
7 IRQ = $0314
```

```
8 BUF = $02C0
9 SID = $D400
10 ENABLE SEI
11 :        LDA #<IRQENTRY
12 :        STA IRQ
13 :        LDA #>IRQENTRY
14 :        STA IRQ+1
15 :        CLI
16 EXIT    RTS
20 IRQENTRY SEI
21 :         LDX #0
22 :         LDA BUF,X
23 :         STA SID,X
24 :         INX
25 :         CPX #$19
26 :         BCC IRQENTRY+3
30 :         JMP $EA31
40 .END
READY.
```

```
0 REM **** P19. ****
1 :
2 :
3 A=828
4 READ P:IF P<0 THEN SYS828:END
5 POKE A,P:A=A+1:GOTO 4
10 DATA 120,169, 73,141, 20,  3,169
11 DATA   3,141, 21,  3, 88, 96,120
12 DATA 162,  0,189,192,  2,157,  0
13 DATA 212,232,224, 25,144,245, 76
14 DATA  49,234, -1
READY.
```

A hang chip egy érdekes alkalmazása – a 6526-os videochip és a kazettás magnetofon összekapcsolása után – a magnetofon jelének a hang chipen keresztül történő átvitele a televízió hangszórójára. Ebben az esetben a magnetofon négyszög jele az I/O elem 13. regiszterének 4. bitjébe impulzusként adódik át. Ezt az impulzust assembler nyelven át kell alakítani egy másik impulzussá („kattanássá") a hang chip számára.

Az impulzus átalakítása többféleképpen történhet. Az egyik módszer szerint (P20, P21 program) az impulzus megjelenésekor a teljes hangerő automatikusan maximum lesz (értéke 15), ami egy kattanás formájában hallható. Majd a program rövid ideig vár és nullázza a hangerőt.

A másik eljárás (P22, P23 program) a kattanás előállítására a kapu jelet használja. Az az impulzus viszont, ami a kapu jel triggerelésekor keletkezik, hangosabb, mint a hangerő kattanása. További lehetőség még, ami a kattanást használja fel, az, hogy a szűrőre vezetjük az oszcillátorokat, majd később leválasztjuk.

A SYS utasítással történő elindítás után bármelyik programmal meghallgatható a kazettán lévő felvétel. Természetesen előzőleg a magnetofont csatlakoztatni kell a géphez. A program leállítása a RUN/STOP RESTORE billentyűkkel kezdeményezett NMI-vel történhet.

A két program azonban nem működik tökéletesen, hisz a négyszög jelekből legfeljebb a beszéd érthető, a zene már nem élvezhető.

```
0 REM **** P20. ****
1 :
2 :
3 OPEN4,4
4 SYS 49152      SYS 36864
5 .OPT OO,P4
6 *=$033C
7 IRQ = $0314
8 BUF = $02C0
9 SID = $D400
10 ENABLE SEI
11 :       LDA #<IRQENTRY
12 :       STA IRQ
13 :       LDA #>IRQENTRY
14 :       STA IRQ+1
15 :       CLI
16 EXIT    RTS
20 IRQENTRY SEI
21 :         LDX #0
22 :         LDA BUF,X
23 :         STA SID,X
24 :         INX
25 :         CPX #$19
26 :         BCC IRQENTRY+3
30 :         JMP $EA31
40 .END
READY.
```

```
0 REM **** P21. ****
1 :
2 :
3 A=828
4 READ P:IF P<0 THEN SYS828:END
5 POKE A,P:A=A+1:GOTO 4
10 DATA 120,169, 73,141, 20,  3,169
11 DATA   3,141, 21,  3, 88, 96,120
12 DATA 162,  0,189,192,  2,157,  0
13 DATA 212,232,224, 25,144,245, 76
14 DATA  49,234, -1
READY.
```

```
0 REM **** P22. ****
1 :
2 :
3 OPEN 1,4
4 SYS 49152  SYS 36864
5 .OPT OO,P
```

```
6 *=$33C
9 ;***** CIMEK
10 KAZETTA = $DC0D
11 SIDREG  = $D400
12 REG1    = SIDREG+4
13 REG2    = SIDREG+11
14 REG3    = SIDREG+18
15 HANGERO = SIDREG+24
16 ;**** KONSTANSOK
17 VARAKOZAS = $F1
18 HULLAM    = $21
19 ;**** PROGRAMKEZDET
20 LDA #15: STA HANGERO
28 ;
29 ;**** VARAKOZAS KAZETTA-IMPULZUSRA
30 CIKLUS  LDA KAZETTA     ;A KAZETTABEMENET KIOLVASASA
31 ↑       CMP #%00010000 ;A BITMINTA OLVASASA
32 ↑       BNE CIKLUS      ;LEKERDEZES,VOLT-E IMPULZUS
38 ;
39 ;**** IMPULZUS BE
40 LDA #HULLAM: STA REG1
41 LDA #HULLAM: STA REG2
42 LDA #HULLAM: STA REG3
48 ;
49 ;**** VARAKOZASI CIKLUS
50 ↑          LDX #0
51 SZAMLALAS  DEX
52 ↑          CPX #CIKLUS
53 ↑          BNE SZAMLALAS
58 ;
59 ;**** IMPULZUS KI
60 LDA #0: STA REG1
61 LDA #0: STA REG2
62 LDA #0: STA REG3
68 ;
69 ;**** VEGTELEN CIKLUS
70 JMP CIKLUS
READY.
```

```
0 REM **** P23. ****
1 :
2 :
3 A=828
4 READ P:IF P<0 THEN SYS 828
5 POKE A,P:A=A+1:GOTO 4
10 DATA 169, 15,141, 24,212,173, 13
11 DATA 220,201, 16,208,249,169, 33
12 DATA 141,  4,212,169, 33,141, 11
13 DATA 212,169, 33,141, 18,212,162
14 DATA   0,202,224,241,208,251,169
15 DATA   0,141,  4,212,169,  0,141
16 DATA  11,212,169,  0,141, 18,212
17 DATA  76, 65,  3, -1
READY.
```

5. A COMMODORE 64-ES ZENEI PROGRAMOZÁSA HALADÓK SZÁMÁRA

5.1. A számláló elv

A zenei programok fajtájuktól függetlenül két részből állnak: a tényleges programból és a hozzá tartozó adatokból.
A tényleges program az adatok felsorolásánál lényegesen rövidebb: nincs más feladata, mint egy hang beolvasása, a hangmagasság beírása a megfelelő regiszterbe, ill. a hang hosszának megfelelő idejű várakoztatása. Az alábbiakban azt fogjuk elemezni, hogyan lehet az ilyen programokat megírni.
Alapvetően ezek a programok két csoportra oszthatók: az egyszólamúakra és a többszólamúakra. Az egyszólamú programok rendkívül egyszerűek. Mindig csak egy hangot kell beolvasni és aztán várakozni. Itt a várakozási ciklus a tényleges lejátszó rutinon kívül helyezkedik el.

Egészen más a helyzet a többszólamú zeneprogramok esetén. Itt felmerül az a probléma, hogy az egyes szólamok hangjai különböző hosszúságúak lehetnek. Ezért a lejátszó rutint és a várakozási ciklust egymásba kell ágyazni.
Ennél a problémánál találkozunk a számláló elvvel, ami a következő: Minden szólam saját számlálóval rendelkezik, mivel a szólamoknak egymástól függetlenül kell futniuk. Ez a számláló lehet inkrementáló (egyesével növekvő) vagy dekrementáló (egyesével csökkenő).

Dekrementáló számláló esetén minden alkalommal, mihelyt egy új hangot kezdünk lejátszani, a hangjegy-időtartam értéke bekerül a számlálóba. A lejátszó rutin minden ciklusa dekrementálódik, a számláló értéke 1-gyel csökken. A folyamatos dekrementálás következtében a számláló értéke valamikor nullázódik. Amint a számláló értéke nulla lesz, egy új hang lejátszása kezdődik és a számláló ismét beállítódik.
A „Michelle" mintaprogramban (P.24) világosan követni tudjuk ezt a folyamatot. Ebben a programban működés közben mindkét számláló értéke kiíródik a képernyőre.

Világosan felismerhető, hogy minden hangjegy elején a számlálóhoz hozzárendelődik egy meghatározott érték, amely mint egy visszaszámláló, mindig dekrementálódik, míg el nem éri a nullát. Mihelyt megjelenik a nulla, egy új hangot játszunk le és az egész kezdődik elölről. Figyeljük meg, hogy a két szólam számlálója egymástól függetlenül fut.

Megjegyzések a programhoz: helyszűke miatt itt eltekinthetünk attól, hogy a hangjegyeket kényelmes módon félhangok és oktávok formájában adjuk meg, mert az átszámító rutin sok helyet és időt foglal el.

A DATA sorok bevitelénél ügyeljünk arra, hogy az első szólam 1000–1007-es, 1009–1016-os és 1038–1045-ös sorai, valamint a második szólam 2000–2006-os, 2007–2013-as és 2030–2035-ös sorai azonosak.
Emiatt bebillentyűzéskor ezeket a sorokat csak egyszer kell beírni, a képernyőn kell hagyni, fel kell lépni a kurzorral és csak a sorszámot módosítva nyomjuk meg a RETURN billentyűt. Ezáltal elég sok beviteli munkát lehet megtakarítani.

Az inkrementáló számláló elve az előzőnek pont a fordítottja: itt a számláló minden ciklus elején nullázódik, majd minden ciklusban eggyel nő. Ezután a számláló tartalma összehasonlítódik a hangjegy időtartamával. Ha a számláló tartalma kisebb, mint a hangjegy időtartama, akkor nem történik semmi. Ellenkező esetben új hangjegy kezdődik és a számláló nullázódik stb.

A harmadik fejezet valamennyi többszólamú programja inkrementálható. Hogy a két elv közül melyiket alkalmazzuk, az a program struktúrájától és hosszától független, mivel mindkettő ugyanazt az eredményt szolgáltatja. A különbség kizárólag annyi, hogy amíg a dekrementáló elvnél a hang időtartamával arányos számmal, addig az inkrementáló elv esetén 0-val hasonlítunk össze, és ez a futási időben árnyalatnyi különbséget okoz.

5.2. Lineáris zeneprogramozás BASIC-ben

Tekintsük a többszólamú lineáris zeneprogram mintapéldájaként a „Michelle" nevű programot (P.24.). A lineáris szó arra utal, hogy a programban az ismétlések nincsenek figyelembe véve. A nem lineáris működtetésű BASIC zeneprogramokat a későbbiek folyamán tárgyaljuk.

A zeneprogram elején a magasabb szintű programnyelvekben megsokallt deklarációk állnak. Assemblerben, mint azt később látni fogjuk, ezeket a deklarációkat a BASIC-hez hasonló módon a forrásszövegben fogjuk megtalálni. Ezzel szemben a futtatható gépi kódban ezek a deklarációk már nem találhatók meg.

Mire kellenek ezek a deklarációk? Már láttuk, hogy a BASIC-ben a hang chip regisztereit POKE utasítással hívhatjuk. Nos, elég fáradságos lenne minden POKE utasítás után ötjegyű számokat bebillentyűzni. Ezért a programban használt hang chip regisztereket változóként kezeljük.

```
0 REM **** P24. ****
1 :
2 :
10 ELSO = 54272
20 L1   = ELSO
30 H1   = ELSO+1
40 W1   = ELSO+4
50 A1   = ELSO+5
60 S1   = ELSO+6
70 L2   = L1  +7
80 H2   = H1  +7
90 W2   = W1  +7
100 A2  = A1  +7
110 S2  = S1  +7
120 POKE A1,3*16+10
130 POKE S1,1*16+7
140 POKE A2,8*16+8
150 POKE S2,8*16+10
200 DIM F1(200), D1(200)
210 DIM F2(200), D2(200)
300 I=0
310 READ F1(I),D1(I):IF F1(I)>=0 THEN I=I+1:GOTO 310
320 I=0
330 READ F2(I),D2(I):IF F2(I)>=0 THEN I=I+1:GOTO 330
400 I1=-1: I2=-1
410 C1=1: C2=1
420 POKE ELSO+24,15
430 PRINTCHR$(147)
500 C1=C1-1:PRINTCHR$(19)"1.SZOLAM:"C1;CHR$(157)"   ",
510 IF C1>0 THEN 580
520 I1=I1+1
530 C1=D1(I1)
540 IF C1<0 THEN POKE L1,0:POKE H1,0:END
550 P2=INT(F1(I1)/256)
560 P1=F1(I1)-P2*256
570 POKE L1,P1:POKE H1,P2:POKE W1,33
580 IF C1<3 THEN POKE W1,32
600 C2=C2-1:PRINT"2.SZOLAM:"C2;CHR$(157)"   "
610 IF C2>0 THEN 670
620 I2=I2+1
630 C2=D2(I2,
640 P2=INT(F2(I2)/256)
650 P1=F2(I2)-P2*256
660 POKE L2,P1:POKE H2,P2:POKE W2,33
670 IF C2<2 THEN POKE W2,32
680 GOTO 500
1000 DATA 14435,12,14435,12
1001 DATA     0, 6,15294, 6
1002 DATA 11457,12,10814, 6
1003 DATA 14435, 6,10814, 6
1004 DATA 10207, 6, 9634, 6
1005 DATA 11457, 6,13625, 6
1006 DATA 11457, 6,10814,12
1007 DATA  9634, 6,11457, 4
1008 DATA 10814,26
1009 DATA 14435,12,14435,12
1010 DATA     0, 6,15294, 6
```

```
1011 DATA 11457,12,10814, 6
1012 DATA 14435, 6,10814, 6
1013 DATA 10207, 6, 9634, 6
1014 DATA 11457, 6,13625, 6
1015 DATA 11457, 6,10814,12
1016 DATA  9634, 6,11457, 4
1017 DATA 10814,20
1018 DATA 14435, 6,19269, 4
1019 DATA 17167, 4,14435, 4
1020 DATA 19269, 4,17167, 4
1021 DATA 14435, 4,21629, 8
1022 DATA 19269,16,    0, 3
1023 DATA 14435, 3,15294, 3
1024 DATA 14435, 3,15294, 6
1025 DATA 11457, 4,11457,26
1026 DATA     0, 3,14435, 3
1027 DATA 14435, 3,14435, 3
1028 DATA 19269, 6,14435, 6
1029 DATA 12860, 3,11457, 3
1030 DATA 10814, 9,10814, 3
1031 DATA 12860, 6,14435, 6
1032 DATA 14435, 6,14435, 6
1033 DATA 14435, 6,14435, 6
1034 DATA 14435, 3,14435,11
1035 DATA 14435, 4,14435,12
1036 DATA 12860, 6,11457, 6
1037 DATA 10814,24
1038 DATA 14435,12,14435,12
1039 DATA     0, 6,15294, 6
1040 DATA 11457,12,10814, 6
1041 DATA 14435, 6,10814, 6
1042 DATA 10207, 6, 9634, 6
1043 DATA 11457, 6,13625, 6
1044 DATA 11457, 6,10814,12
1045 DATA  9634, 6,11457, 4
1046 DATA 10814,20, 9634, 3
1047 DATA 10814, 3,11457, 6
1048 DATA  9634, 6,12860, 6
1049 DATA 10814, 6,11457, 6
1050 DATA  9634, 6,12860, 9
1051 DATA 10814, 3,11457,12
1052 DATA 10814, 6, 9634, 6
1053 DATA  9094,12, 9634, 6
1054 DATA 10814, 6, 9634,96
1055 DATA  9094,24
1999 DATA -1,-1
2000 DATA  2408,12, 3608,12
2001 DATA  3215,12, 3823,12
2002 DATA  4291,18, 4050, 6
2003 DATA  3823,24, 3608,12
2004 DATA  3823,12, 3608, 6
2005 DATA  5407, 6, 3608, 6
2006 DATA  5407, 6
2007 DATA  2408,12, 3608,12
2008 DATA  3215,12, 3823,12
2009 DATA  4291,18, 4050, 6
2010 DATA  3823,24, 3608,12
2011 DATA  3823,12, 3608, 6
2012 DATA  5407, 6, 3608, 6
2013 DATA  5407, 6
2014 DATA  2408,12, 2408,12
2015 DATA  2408, 6, 3608, 6
2016 DATA  4817, 6, 5728, 6
2017 DATA  4291,12, 2864,12
2018 DATA  3823,12, 4291, 6
2019 DATA  4817, 6, 7217,12
2020 DATA  4817,12, 3823,12
2021 DATA  3215,12
2022 DATA  2408, 6, 3608, 6
2023 DATA  2273, 6, 3608, 6
2024 DATA  2145, 6, 3608, 6
2025 DATA  2025, 6, 3608, 6
2026 DATA  1911, 6, 3215, 6
2027 DATA  1911, 6, 2864, 6
2028 DATA  1804, 6, 2703, 6
2029 DATA  1804, 6, 2703, 6
2030 DATA  2408,12, 3608,12
2031 DATA  3215,12, 3823,12
2032 DATA  4291,18, 4050, 6
2033 DATA  3823,24, 3608,12
2034 DATA  3823,12, 3608, 6
2035 DATA  5407, 6, 3608, 6
2036 DATA     0, 6
2037 DATA  3823,24, 3608,24
2038 DATA  3215,24, 3608,24
2039 DATA  4817, 8, 4817, 2
2040 DATA  5407, 2, 6069, 8
2041 DATA  6069, 2, 7217, 2
2042 DATA  7647,12, 5728,12
2043 DATA  5407, 8, 5407, 2
2044 DATA  5728, 2, 6430, 6
2045 DATA  5407, 6, 4817, 8
2046 DATA  4817, 2, 5407, 2
2047 DATA  5728, 6, 4817, 6
2048 DATA  3608,24
2999 DATA -1,-1
READY.
```

Az alábbi táblázatban a programban előforduló valamennyi változót megtaláljuk.

VÁLTOZÓ	FUNKCIÓ
L1	1. oszcillátor frekvencia alsó byte
H1	1. oszcillátor frekvencia felső byte
W1	1. oszcillátor hullámforma és kapujel
A1	1. oszcillátor AD értéke
B1	1. oszcillátor SR értéke

(az L2, H2, ... stb. változók a 2. oszcillátor megfelelői)

C1	1. szólam számlálója
I1	1. a szólam indul
F1	Első szólam frekvencia tömbje (vektor)
D1	Első szólam időtartam tömbje (vektor)

(az I2, C2 ... stb. változók a 2. oszcillátor megfelelői)

(P1, P2 az alsó–felső átszámításához kell, I pedig a tömbindex)

Könnyen felismerhető, hogy a programban alkalmazott változókat három csoportba lehet osztani:

1. A hang chip regiszter változói
2. A hangmagassági és időtartam adatokat érintő változók
3. Az átszámításokra és egyéb feladatokra használt változók

Az előbbi zenei program működése ezen felosztás alapján egy mondatban így foglalható össze: a 2. változócsoport értékei a 3. változócsoport segítségével meghatározott időközönként átadódnak az 1. változócsoportnak.

Ezt a folyamatot a „hang chip regisztereinek rendszeres felülírásával" is jellemezhetjük. Egy zenei program feladata ennél se több, se kevesebb. Meghatározott adatokat (hangmagasság) meghatározott időközönként megszakítva (hang időtartam) be kell írni a hang chip regisztereibe.

Most nézzük tovább a programot. A deklarációk után egy olyan programszakasz következik, melyben a második csoport változóihoz meghatározott kezdőértékeket rendelünk hozzá.

Ezt a folyamatot, a kezdőérték hozzárendelésének folyamatát *inicializálás*nak nevezzük. Programunkban négy tömböt dimenzionálunk és feltöltjük a DATA utasítás adataival. Ezután feltöltjük a számlálókat a kezdeti értékekkel.

Az inicializálás után az 500–680-as sorokban találjuk a zeneprogram magját, a lejátszórutint. A rutin a négy tömb tartalmát értékeli ki. A hangmagasságokat és hangidőtartamokat lejátssza, miközben a tömbindexet időről időre inkrementálja.

Ha az inicializálás közben az indexet a tömb végére állítottuk és a lejátszó rutin ezt dekrementálja, akkor a zenedarab visszafelé is lejátszható. A fentiek alapján látható, hogy a két független számláló mellett még szükség van két független tömbindexre is.
A számláló és az index együttesen határozzák meg a szólamok időbeli lefutását. A számláló értéke attól függ, hogy milyen hosszan kell lejátszani egy hangjegyet. A DATA sorokban a hang időtartama alatt mindig azt az értéket találjuk, amelyet a számláló egy hangjegy lejátszása alatt maximálisan felvehet.

A hangjegyindex értéke egy szólamon belül mindig a hangjegyek számától függ. Az index működésének módját úgy tehetjük szemléletesebbé, ha egy szólam összes hangjegyét beszámozzuk. Az index egyszerűen az első hangjegytől az utolsóig számol.

A „Michelle" program egy lineáris zeneprogram. Bevitelkor láttuk, hogy a mű egy része háromszor is előfordul és ezért azt háromszor kell bebillentyűzni.

Ha megértettük az index működési elvét, és programozástechnikailag alkalmazni tudjuk, akkor egy zenedarab bevitelénél az index-vezérlés kibővítésével munkát takaríthatunk meg.

Eddigi példáinkban az index mindig csak eggyel nőtt (lineárisan). A következő programban nézzük meg, hogy az index visszaállításával hogyan lehet az ismétlést vezérelni.

```
0 REM **** P25. ****
1 :
2 :
10 ELSO = 54272
20 L1   = ELSO
30 H1   = ELSO+1
40 W1   = ELSO+4
50 A1   = ELSO+5
60 S1   = ELSO+6
70 L2   = L1  +7
80 H2   = H1  +7
90 W2   = W1  +7
100 A2  = A1  +7
110 S2  = S1  +7
120 POKE A1,6*16+7
130 POKE S1,6*16+12
140 POKE A2,1*16+6
150 POKE S2,8*16+7
200 DIM F1(200), D1(200)
210 DIM F2(200), D2(200)
```

```
300 I=0
310 READ F1(I),D1(I):IF F1(I)>=0 THEN I=I+1:GOTO 310
320 I=0
330 READ F2(I),D2(I):IF F2(I)>=0 THEN I=I+1:GOTO 330
400 I1=-1:I2=-1
410 C1=1:C2=1
420 TI$="000000"
430 T1=TI:T2=T1
440 POKE ELSO+24,15
500 IF T1<TI+100 THEN C1=C1-1
510 IF C1>0 THEN 580
520 I1=I1+1
530 C1=D1(I1):T1=TI
540 IF C1<0 THEN POKE L1,0:POKE H1,0:END
550 P2=INT(F1(I1)/256)
560 P1=F1(I1)-P2*256
570 POKE L1,P1:POKE H1,P2:POKE W1,33
580 IF C1<5 THEN POKE W1,32
600 IF T2<TI+100 THEN C2=C2-1
610 IF C2>0 THEN 670
620 I2=I2+1
630 C2=D2(I2):T2=TI
640 P2=INT(F2(I2)/256)
650 P1=F2(I2)-P2*256
660 POKE L2,P1:POKE H2,P2:POKE W2,33
670 IF C2<2 THEN POKE W2,32
680 IF I2=31 THEN IF C2=1 THEN I2=-1:C2=1
690 GOTO 500
1000 DATA  8583,24, 7647,24
1001 DATA  7217,24, 8101,24
1002 DATA  8583,24,10207,24
1003 DATA 11457,24,12860,24
1004 DATA     0,24,     0,24
1005 DATA     0,24,     0,24
1006 DATA  8583,18, 6430, 6
1007 DATA 10207,18, 7647, 6
1008 DATA 11457,18, 8583, 6
1009 DATA 12860,18, 9634, 6
1010 DATA  8583,24,     0,24
1011 DATA     0,24,     0,24
1012 DATA 17167, 6,12860, 6
1013 DATA 11457, 6,12860, 6
1014 DATA  8583,12,     0,12
1015 DATA  8583, 6,11457, 6
1016 DATA 12860, 6,17167, 6
1017 DATA 16203,12,     0,12
1018 DATA  8583, 6
1999 DATA -1,-1
2000 DATA  1072, 3, 1072, 3
2001 DATA   803, 3,  955, 3
2002 DATA  1072, 3, 1072, 3
2003 DATA   803, 3,  955, 3
2004 DATA  1275, 3, 1275, 3
2005 DATA   955, 3, 1136, 3
2006 DATA  1275, 3, 1275, 3
2007 DATA   955, 3, 1136, 3
2008 DATA  1432, 3, 1432, 3
```

```
2009 DATA  1072, 3, 1275, 3
2010 DATA  1432, 3, 1432, 3
2011 DATA  1072, 3, 1276, 3
2012 DATA  1607, 3, 1607, 3
2013 DATA  1204, 3, 1432, 3
2014 DATA  1607, 3, 1607, 3
2015 DATA  1204, 3, 1432, 3
2999 DATA -1,-1
READY.
```

5.3. Példa nemlineáris BASIC zeneprogramra

Az előző oldalakon szereplő program bevitele toolkit segítségével leegyszerűsíthető. A DELETE 1000– utasítás után a főprogramot nem kell újra bevinni. Elég, ha a két lista összehasonlítása után csak a különbséget visszük be a memóriában lévő programba, szerkesztés (editálás) segítségével.

Ez a program egy olyan dallamot tartalmaz, amely Donna Summer „I feel love" című dalának egésze alatt hallható. Az ilyen basszuskíséretet, amely nagyobb időközönként mindig ismétlődik, ostinato basszusnak, vagy egyszerűen ostinatonak hívják.

Itt az ismétlés legegyszerűbb módjával állunk szemben. Az egyik szólam egy szakasza állandóan ismétlődik. Az ismétlés azonban csak a második szólamban fut, az első szólam teljesen független a másodiktól.

Ha ezt az ismétlést nem egy programrésszel vezérelnénk, akkor a 2000–2015-ös sorokat ismétlésenként újra és újra be kellene vinni. Ez szükségtelenül megnövelné a programot. E sok felesleges munka helyett a 680-as sorban van egy utasítás, amely az ismétléseket vezérli.
Ez az utasítás a következőképpen is értelmezhető: ha az (I2) index elérte az ismétlés előtti utolsó hangjegy értékét, megvizsgálja, hogy a (C2) számláló a hangjegy végét mutatja-e.

Ha igen, akkor az indexet az ismétlendő programrész előtti értékre állítja, a számlálót pedig arra az értékre, amely a lejátszó rutinban a futás során következik.

Ez az adott esetben egy elég bonyolult utasításláncot eredményez, melynek bevitelekor sok hibát követhetünk el. De ez még mindig könnyebb, mint számunkra értelmetlen DATA sorokat többször bebillentyűzni, ahol szintén nagy a tévesztés veszélye.

Az IF ... THEN IF ... THEN IF ... THEN utasításoknak ezt a módját találjuk a harmadik fejezet programjaiban is, melyekben ismétlést programoztunk.

Egy megjegyzés a BASIC nyelvet ismerőknek: zenei programokban (de természetesen máshol is) időt takaríthatunk meg az IF ... THEN IF ... THEN utasításlánc használatával. Ugyanis az IF(...AND ... AND ... AND ...) THEN változat futása közben minden ciklusban az összes feltétel megvizsgálódik, ami nagyon sok időt vesz igénybe. Ha tehát az IF...THEN utasításokat ügyes sorrendben ágyazzuk egymásba, akkor a feltételek különböző prioritása miatt igen sok időt takaríthatunk meg.

5.4. Korlátok a BASIC nyelvű zeneprogramozásban

Az egyszólamú zeneprogramok jól programozhatók BASIC nyelven. A többszólamú zeneprogramoknál az a probléma, hogy bizonyos sebesség fölé nem léphetnek. A zenemű sebessége, tempója nem is annyira fontos, mint az egyidejűleg játszandó hangjegyek követési ideje.

Az természetes, hogy két hangot „hajszálpontosan" egy időben sem egy zongorista, sem egy számítógép nem tud egyszerre megszólaltatni. Ez fizikai képtelenség. A kérdés csupán az, hogy milyen gyorsan lehet egymásután két hangot lejátszani úgy, hogy azt a benyomást keltse, mintha valójában egyidejűek lennének.

Az emberi fül kb. 0,1 másodperc időeltolással még felismeri a két hangot. Azok a hangok, amelyek ennél gyorsabban követik egymást, egyidejűleg hallatszanak. A zenei programok feladata az, hogy ez alatt az időkorlát alatt is meg tudják szólaltatni a hangokat. Pontosan emiatt a többszólamú programok nem szólaltathatók meg BASIC nyelven.

Kétszólamú programoknál még nincs olyan nagy gond, azonban a háromszólamúaknál már érezhető a pontosság csökkenése.
Egy többszólamú zeneprogram lehetőleg pontos hangzásához az szükséges, hogy az egyes szólamok hangjait a lehető leggyorsabban játsszuk le egymás után. Tehát a rutinoknak minél rövidebbnek kell lenniük és közvetlenül egymás után kell elhelyezkedniük.

Rendelkezésünkre állnak olyan programozási fogások, melyek segítségével ezeket a rutinokat úgy lehet megírni, hogy azok időben lehetőleg a leggyorsabban fussanak le. Azt a fogást már láttuk, hogyan lehet több szűrő lekérdezését IF... THEN IF ... THEN utasításlánccal időtakarékosan programozni.

Most ismerkedjünk meg a frekvenciák felső, ill. alsó értékekre való átszámításának módszerével.
Az előző mintaprogramban szándékosan olyan módszert választottunk, amellyel időt lehet veszíteni. Figyeljük meg az 550–570-es, és a megfelelő 640–660-as

sorokat! Láthatjuk, hogy az átszámítás a lejátszó rutinon belül történik. Ez az oka annak, hogy ebben a programban olykor úgy tűnik, a hangjegyek nem a megfelelő sorrendben követik egymást.

A második fejezet programjaiban, melyekben a BASIC nyelvű zeneprogramozás legjobb módszerét mutattuk be, ezt az átszámítást a tényleges lejátszó rutinon kívül helyeztük el.

5.5. Egy megjegyzés a kapcsoló elvhez

Láttuk, hogy egy szólam indexének visszaállításánál ismétlést lehet programozni. Ennél az eljárásnál fontos volt, hogy az index arra a hangjegyre mutasson, amelyikkel az ismétlést kezdeni kellett, mert első híváskor a lejátszó rutin az ismétlés első hangjegyét játssza.

Ha egy zenemű meghatározott részét többször szeretnénk ismételni, akkor szükség van egy további változó bevezetésére, mely az ismétléseket számlálja. Ezt a változót könyvünkben *kapcsoló*nak nevezzük.

Minden szólamnak külön kapcsolója van. Ezeket a program elején nullázni kell. Minden ismétlés végén az aktuális szólam kapcsolója eggyel növekszik. Ezután megkérdezi, hogy az ismétlő rutin a maximális értékét elérte-e. Igen válasz esetén a program az ismétlő rutint átugorja.

A kapcsolók alkalmazását a második fejezetben szereplő példákon keresztül mutatjuk be. A „Yesterday" programban a kapcsolókat ismétlésre és tempó vezérlésre is használjuk. A „Strawberry Fields Forever" programban arra mutatunk példát, hogyan lehet a kapcsolókat a hangzás átkapcsolására alkalmazni. Az assembler nyelvben történő programozásnál további alkalmazásokat láthatunk a kapcsoló elvre.

5.6. A zenei segédlet elve

Miután megismerkedtünk a BASIC nyelvű zenei programozás gyenge pontjaival, nézzük meg, hogyan dolgozhatunk jobban. Mivel a BASIC nyelv fájó pontja a sebesség, könnyen adódik a megoldás, hogy a gépi kódot vagy assemblert használjunk. Assemblerben a sebesség már nem jelent problémát. Ehelyett más, még nehezebb gonddal kell szembenéznünk: a kezelhetőséggel. A monitorral és assemblerrel való munka egy nem profi számára komoly megerőltetést jelent. Ezért szorítkoztunk eddig a BASIC nyelvre, mivel ott a programkészítési lehetőségek optimálisak.

Mivel a BASIC nyelvnek és az assemblernek egyaránt megvan a maga előnye és hátránya, célszerűnek látszik egy olyan eljárás alkalmazása, amely a két programozási nyelv előnyeit összevonja. Ezt az eljárást nevezzük zenei segédletnek.

Azokon a helyeken, ahol a sebesség lényeges, a már jól ismert lejátszó rutint assembler nyelven írjuk meg, így jelentősen megnöveljük a sebességet. A hangjegy bevételét a kényelmesebb BASIC nyelven programozzuk. Ez a legtöbb zenei segédlet lényege. Ezek a segédletek olyan BASIC-hez hasonló kiegészítő utasításokat tartalmaznak, amelyek egyidejűleg több hangjegyet tudnak lejátszani. A hangjegyeket BASIC utasítás formájában lehet beadni.

Jelenleg több különféle zenei segédlet létezik már, és várható, hogy számuk még nőni fog. Ezért egy konkrét zenei segédlet részletes leírására a helyszűke miatt nem vállalkozunk. Itt csak az alapjait kívánjuk megmutatni ennek a munkának.

A legtöbb segédlet lehetővé teszi egy zenemű hangjegyeinek névvel és oktávval való bevitelét (például a második fejezet hosszabb programjai esetén). Ennél az eljárásnál már nem kell segítségül hívni állandóan a negyedik fejezetben lévő hangfrekvencia táblázatot.

A hangjegy átszámítás assemblerben történik és ennélfogva rövidebb, mint egy tiszta BASIC program. A segédletek alkalmazásakor a hangjegyek jelölésére kell nagyon ügyelni.
Az angol nyelvterületről származó programok a „H" helyett a „B" jelölést használják.
Továbbá ügyelni kell az emelések, illetve leszállítások jelölésére, azaz arra, hogy milyen formában kell pl. a „Cisz" és „Desz" hangokat bevinni. Ezenkívül nincs szabványosítva az oktáv jelölése. Általában – mint ebben a könyvben is – a legmélyebb oktáv jele 0 (pl. C0), a legmagasabb oktávé pedig 7 (pl. A7).
A következő lényeges kérdés a hangjegy hangzásának időtartama. Míg a hangjegynév esetén a legtöbb zenei segédlet egységes, addig a hangjegy hangzásának időtartamára minden szerző más és más beviteli formátumot talál ki.

Sok zenei segédlet használja a zenei negyed és fél hangjegy jelölést, azonban a beviteli formátumuk erősen különbözik egymástól. A „Synthy 64" törtvonalat, a „SIMON'S BASIC" egy funkcióbillentyűt használ. Itt sajnos nincs mit tenni, utána kell nézni a használati utasításban.
Miután elvégeztük a hangjegy bevitelét, a hangszín beállítása következik. Ezen a ponton térnek el a zenei segédletek egymástól a leginkább, és itt lehet legjobban megítélni a minőségüket.

A legtöbb zenei segédlet ezen a téren ötletességgel jellemezhető. Van egy utasításuk (mint a SIMON'S BASIC "SOUND" utasítása), amely a hullámforma beállítását úgy biztosítja, mint a POKE utasítás. Ezzel valójában a felhasználó nem nyer,

hiszen ugyanúgy használhatja a POKE utasítást is. Ugyanez a helyzet a szűrőbeállításnál is. Itt a biteket ugyanúgy a felhasználónak kell kiszámítania, mint a POKE utasítás esetében.

Egyedül a lejátszó rutin az, amellyel a felhasználónak nem kell törődnie. A kapu jel automatikus beállítása és törlése a zeneművek lejátszása közben minden lejátszó rutinnál azonos.

A „Synthy 64" esetében azonban van egy különbség, mert ennek a zenei segédletnek van egy ún. TRACE funkciója.
Ha megnézzük még egyszer a negyedik fejezet „Read – Write Register" programját, az ott alkalmazott elv az alapja ennek a funkciónak is. Ugyanis valamennyi regiszter tartalmát egy átmeneti tárolóban helyezzük el, így ezeket bármikor le lehet kérdezni.

Az átmeneti tároló segítségével lépésről lépésre lehet követni, hogy egy mű megszólalása közben a különböző hangjegyek hogyan játszódnak le, és még a hang-átkapcsolások is figyelemmel kísérhetők. Példa lehet más zenei segédletek szerzői számára a „Synthy 64" hang-átkapcsolása; itt BASIC alprogramként lehet bevinni a hang-beállításokat és ezeket GOSUB segítségével lehet hívni.
A zeneművek futtatásának vezérlésére gyakran van egy további utasítás is, amelylyel a futási sebességet lehet tetszés szerint változtatni (pl. TEMPO, SPEED stb.). Ebben az esetben lényegesen nagyobb sebességi tartalék áll rendelkezésre, mint a BASIC zenei programoknál.

Mindezeket – a nagyobb sebesség kivételével – BASIC rutinokkal is lehet szimulálni. Van azonban egy pont, ahol ez nem lehetséges, és ezért bizonyos zenei segédletek döntőnek bizonyuló előnnyel rendelkeznek.

Vannak segédletek, amelyek egy másik program futása alatt is tudnak zenét szolgáltatni. Azaz amíg a gép egy zeneművet játszik, addig ezzel egy időben egy másik program részei is futhatnak, mint pl. sprite vezérlés, grafikai rutin, joystick lekérdezés.

Azonban ez is csak korlátozott keretek között történhet.
Ennek az az oka, hogy mihelyt valamilyen úton-módon egy programsorban a változók kezelésére, vagy egy parancs keresésére BASIC rutint kell alkalmazni, rengeteg időt veszítünk.
Ezek a rutinok ugyanis nagyon időigényesek.

Ha játékot kell programoznunk, ahol a grafikát és a zenét valamilyen módon kombinálni kell, bizonyára nem gondolunk arra, hogy a SIMON'S BASIC segítségével oldjuk ezt meg.
Azt a sebességet, amelyet egy játékprogram igényel, csak akkor érhetjük el, ha

teljes mértékben assemblerben programozunk. Akkor sem tudjuk az assemblert kikerülni, ha csak egy címzenét akarunk írni valamelyik programunkhoz. Ezért nézzük most meg a zenei programozást assemblerben.

5.7. Megjegyzések a könyv assembler példaprogramjaihoz

Ebben a könyvben az összes assembler programot a „Profi–Ass" segítségével hoztuk létre. Ezt az assemblert tartalmazza a PROFIMAT programcsomag is. Az a Profi–Ass változat, amellyel a programok készültek, a $C000–$CFFF tartományban helyezkedik el és így nem foglal el BASIC-RAM területet. Ha olyan Profi–Ass változattal dolgozunk, amely a $9000–$9FFF területen helyezkedik el, ez 4K BASIC-RAM területet foglal el. Figyelni kell azonban arra, hogy a minden Profi–Ass forrásszöveg kezdetén szereplő SYS 49152 utasítást SYS 36864-re kell módosítani.

5.8. Egy többszólamú assembler zeneprogram

Nagyon sok helyet foglalna el, ha a harmadik fejezethez hasonlóan most is végig követnénk egy assembler program fejlesztését az egyszólamútól a többszólamúig. Ráadásul a programozástechnikai összefüggések is összehasonlíthatatlanul bonyolultabbak. Ennélfogva egy kész háromszólamú programot mutatunk be.

A Genesis együttes „Cinema Show" című számáról van szó. Mint a könyv többi assembler programjánál, itt is megadjuk a forrásprogramot és a BASIC betöltőt a darab meghallgatásához. Ha rendelkezünk programmal, akkor ez jó gyakorlásnak bizonyul a forrásprogram bevitelére, mivel megtanulhatunk néhány dolgot az assembler programok szerkezetéről.

A továbbiak során lépésről lépésre végighaladunk a forrásprogramon. A programrészleteket minden különösebb jelölés alkalmazása nélkül állandónak kell tekinteni. Megjelöltük azonban a változtatható programrészleteket. Minden további nélkül meg lehet változtatni a programot úgy, hogy más zeneművet játsszon.

0003–0005: Mivel a gépi kódú program $8000-tól fölfelé helyezkedik el, ezért a BASIC-RAM felső határát $8000-re kell állítani.

A SYS-szel történő indítás után a kiviteli opciók „00"-ra állnak be.

0013–0037: A deklarációk első része. Deklarálódik valamennyi hang chip regiszter.

0050–0073: A deklaráció második részében valamennyi hangjegyet a Profi-Ass szimbólumaként határozunk meg. Ennek az az előnye, hogy a hangjegy átszámítás a gépi kóddá való átalakítás során történik.

0080–0089: Az assembler program „változóinak" definiálása történik a harmadik deklarációs részben. Itt a számláló (counter), a mutató (pointer) és a kapcsolók (flag) meghatározásáról van szó.

0110–0118: Szövegkiadó rutin a címszöveghez.

0120–0149: A címszöveg ASCII kódjainak szabad területe, ami változtatható a beírandó szöveg nagysága szerint.

0150–0152: A tényleges lejátszó program szubrutinként kerül behívásra, melynek végén a hangerő nullázódik, majd az RTS-sel visszatérhetünk a BASIC-be.

0200–0250: Ez az inicializáló programrész, mely fix és változó assembler utasítások keveréke. Lényeges, hogy az assembler program változói, vagyis a számláló és a mutató, definiált kezdőértékek legyenek. Az inicializálással állítjuk be a kezdőhangot is. Ügyeljünk arra, hogy a három oszcillátor hullámformáját a WAVE1, WAVE2 és a WAVE3 változóban kell tárolni. A kapu jel vezérlése érdekében az LSB-t törölni kell.

0500–0523: Ez az első szólam lejátszó rutinja, mely függetlenül fut a másik két szólamtól. Ezt egy számlálóval oldottuk meg, amely értéke mindig 1-gyel nő. Ha a számláló elérte a hang időtartamának értékét, akkor egy új hang kezdődik és a kapu jel visszaállítódik.

0600–0623: Ez a második szólam lejátszó rutinja, melyre ugyanaz vonatkozik, mint az első szólamra.

0700–0723: A harmadik szólam lejátszó rutinja, melyre szintén ugyanaz vonatkozik, mint az első szólamra.

0800–0918: Ez egy szabad terület tetszés szerinti kapcsoló kérdezésére. Kapcsoló lekérdezésével egy szólam tetszés szerinti részét kívánság szerint meg lehet ismételni, és a mű tetszés szerinti helyén hangátkapcsolást lehet végezni. Az i-edik kapcsoló lekérdezésének formátuma az alábbi:

```
XX0 N<i−1> * = *
XX1 LDA POINTER <V> :CMP#<POS<nr> −1:BNE N<i>
XX2 LDA POINTER <V>+1:CMP#>POS<nr> −1:BNE N<i>
XX3 LDY #0: LDA (POINTER<V>), Y
XX4 TAX:DEX : CPX COUNTER <V> :BNE N<i>
```

A program egy meghatározott időpontban hajtja végre ezt az utasítás sorozatot. Vagyis akkor, amikor a POS⟨nr⟩ címke előtti hangjegy lejátszását befejezte. Ettől eltérő esetekben a BNE a következő kapcsoló lekérdezésére léptet; tehát akkor, ha a mutató nem az ismétlés előtti utolsó hangjegyre mutat (ami akkor áll fenn, amikor a számláló nem érte el a maximális értéknél eggyel kisebb értéket).

(Most láthatjuk, hogy az assembler programban az inkrementáló számláló elvét alkalmazzuk, mivel a számláló tartalma a maximális értékkel kerül összehasonlításra.)
Ismétléskor az alábbi utasításlánc végrehajtása történik, folytatásként:

```
XX5 INC FLAG ⟨V⟩
XX6 LDA FLAG ⟨V⟩: CMP#⟨AZ UTOLSÓ ISMÉTLÉS UTÁNI ÉRTÉK + AZ IS-
                  MÉTLÉSEK SZÁMA⟩: BEQ N⟨i⟩
XX7 LDA # < ⟨AZ ÚJRAKEZDÉS POZÍCIÓJA⟩: STA POINTER⟨V⟩
XX8 LDA # > ⟨AZ ÚJRAKEZDÉS POZÍCIÓJA⟩: STA POINTER⟨V⟩ +1
```

⟨AZ ÚJRAKEZDÉS POZÍCIÓJA⟩ adatrészben az XXXX POS ⟨nr⟩ .BYTE 1 formátumnak kell szerepelni. A ".BYTE 1" elhagyható, az alábbi formátum alkalmazása esetén:

(Az XX5 és XX6 sorok megegyeznek)

```
XX7 LDA # < ⟨AZ ÚJRAKEZDÉS POZÍCIÓJA⟩ –1 :STA POINTER ⟨V⟩
XX8 LDA # > ⟨AZ ÚJRAKEZDÉS POZÍCIÓJA⟩ –1 :STA POINTER ⟨V⟩+1
XX9 LDA # 200 : STA COUNTER ⟨V⟩
```

Ezután a pozíciósor formátuma hagyományos: XXXX POS ⟨nr⟩ *=*
Az *=* a Profi-Ass üres utasítása. Abban az esetben használható jól, ha egy programsorban csak egy címkét akarunk definiálni.

Az előző sorokban XXX mindig egy sorszámot vagy annak egy részét jelentette. Az ⟨i⟩ a kapcsoló lekérdezések számát jelölte. A ⟨V⟩ azt a szólamot jelenti, amelyikre a kapcsoló lekérdezése vonatkozott. Az ⟨nr⟩ a pozíciósor száma, melyet a jobb érthetőség kedvéért az alábbiak szerint kell beállítani:

POS11, POS12, POS13 ..... Az első szólam pozíciósora
POS21, POS22, POS23 ..... A második szólam pozíciósora
POS31, POS32, POS33 ..... A harmadik szólam pozíciósora

Ha hangszín átkapcsolást akarunk programozni, akkor az XX5–XX9 sorokat a hang chip regisztereinek beállítására lehet használni. A hullámforma módosítása esetén változtatnunk kell a WAVE1, WAVE2, WAVE3 változókat. Ráadásként a SPEED változóhoz is új érték rendelhető. Ezzel lehetővé válik a zeneművön belül a tempó módosítása.

0990–0992: Ez a lejátszó rutin és a kapcsoló lekérdezés minden lefutása után végrehajtódó késleltető ciklus. Lehetőség van továbbá a LOOP ciklus és a JMP LOOP 1 ugró utasítás között egy más programrészlet tárolására. Egy gyakorlott programozó közbeszúrhat pl. egy grafikai rutint. Ez a beszúrt rutin eleget kell tegyen annak a feltételnek, hogy állandó ciklusidővel rendelkezzén. Szükséges tehát, hogy ez a rutin csak rövid időt vegyen igénybe, és ezután rögtön a JMP LOOP 1 utasítás végrehajtása következzék. Nem szabad pl. egy olyan rutint elhelyezni, mely egy billentyű lenyomására vár, tehát maga is várakozási ciklussal rendelkezik.

1000–1999: Változtatható adatrész, mely az első szólam hangmagasságait és hang időtartamait tartalmazza. A 1000-es sor kötelező. Az összes többi vagy pozíciósor, vagy az alábbi felépítésű: XXXX. WORD ⟨HANGMAGASSÁG⟩: .BYTE ⟨HANG IDŐTARTAM⟩. A hang időtartam esetében a megengedett érték 0–255 lehet. Hangmagasságként 0 és 65535 között minden érték megengedett. A 0 szünetként alkalmazható. Egyébként pedig azokat a hangjegy szimbólumokat lehet használni, amelyeket az 50–73-as sorokban definiáltunk.

A 2000–2999-es és 3000–3999-es sorok a második és a harmadik szólam hangmagasságait és hang időtartamait tartalmazzák. A 2000-es és a 3000-es sornak itt is

⟨sorszám⟩ SZÓLAM ⟨X⟩ .BYTE 1

formátumúnak kell lennie, ahol ⟨X⟩ a szólam száma. A továbbiakban minden az első szólammal megegyezőképpen történik.

```
0 REM **** P26. ****
1 :
2 :
3 POKE 56,128:CLR:SYS 49152
4 .OPT 00
5 * =$8000
6 ;
7 ; THE CINEMA SHOW
8 ; ASSEMBLER FORRASPROGRAM
9 ; THOMAS DACHSEL FELDOLGOZASA
10 ;
11 ; >>KONSTANSOK
12 ;
13 OSC1LF = $D400
14 OSC1HF = $D401
15 OSC1PL = $D402
16 OSC1PH = $D403
17 OSC1WV = $D404
18 OSC1AD = $D405
19 OSC1SR = $D406
20 OSC2LF = $D407
21 OSC2HF = $D408
22 OSC2PL = $D409
23 OSC2PH = $D40A
24 OSC2WV = $D40B
25 OSC2AD = $D40C
```

```
26 OSC2SR = $D40D
27 OSC2LF = $D40E
28 OSC2HF = $D40F
29 OSC3PL = $D410
30 OSC3PH = $D411
31 OSC3WV = $D412
32 OSC3AD = $D413
33 OSC3SR = $D414
34 FILTLF = $D415
35 FILTHF = $D416
36 FILMOD = $D417
37 VOLUME = $D418
50 C1   =   536:C2   =  1072:C3   = 2145
51 C4   =  4291:C5   =  8583:C6   =17167
52 CS1  =   568:CS2  =  1136:CS3  = 2273
53 CS4  =  4547:CS5  =  9094:CS6  =18188
54 D1   =   602:D2   =  1204:D3   = 2408
55 D4   =  4817:D5   =  9634:D6   =19269
56 DS1  =   637:DS2  =  1275:DS3  = 2551
57 DS4  =  5103:DS5  =10207:DS6  =20415
58 E1   =   675:E2   =  1351:E3   = 2703
59 E4   =  5407:E5   =10814:E6   =21629
60 F1   =   716:F2   =  1432:F3   = 2864
61 F4   =  5728:F5   =11457:F6   =22915
62 FS1  =   758:FS2  =  1517:FS3  = 3034
63 FS4  =  6069:FS5  =12139:FS6  =24278
64 G1   =   803:G2   =  1607:G3   = 3215
65 G4   =  6430:G5   =12860:G6   =25721
66 GS1  =   851:GS2  =  1703:GS3  = 3406
67 GS4  =  6812:GS5  =13625:GS6  =27251
68 A1   =   902:A2   =  1804:A3   = 3608
69 A4   =  7217:A5   =14435:A6   =28871
70 B1   =   955:B2   =  1911:B3   = 3823
71 B4   =  7647:B5   =15294:B6   =30588
72 H1   =  1012:H2   =  2025:H3   = 4050
73 H4   =  8101:H5   =16203:H6   =32407
77 ;
78 ;>>VALTOZOK
79 ;
80 POINTER1 =2:POINTER2 =5:POINTER3 =8
81 COUNTER1 =4:COUNTER2 =7:COUNTER3 =10
82 WAVE1    =$2C0
83 WAVE2    =$2C1
84 WAVE3    =$2C2
85 FLAG1    =$2D0
86 FLAG2    =$2D1
87 FLAG3    =$2D2
88 SPEED    =$2E0
89 FLAGX    =$2E1
100 ;
101 ;>>FOPROGRAM
102 ;
110 LDY #0
111 LDA #<TEXT:STA POINTER1
112 LDA #>TEXT:STA POINTER1+1
113 READSIGN LDA (POINTER1),Y
114 BNE PRINT
```

```
115 JMP PLAY
116 PRINT JSR $FFD2
117 INC POINTER1:BNE *+4:INC POINTER1+1
118 JMP READSIGN
120 TEXT  .BYTE $93,$D,$D,$D,$D,$D,$D,$D,$D,$D,$D
121 .ASC "                THE CINEMA SHOW":.BYTE $D
122 .ASC "               MUSIC BY: GENESIS":.BYTE $D
123 .ASC "      ARRANGEMENT BY: THOMAS DACHSEL"
124 .BYTE 0
150 PLAY SEI:JSR INIT:CLI
151 LDA #0:STA VOLUME
152 RTS
200 INIT LDA # 31:STA VOLUME
201 ↑     LDA #$25:STA OSC1AD
202 ↑     LDA #$7B:STA OSC1SR
203 ↑     LDA #$25:STA OSC2AD
204 ↑     LDA #$7B:STA OSC2SR
205 ↑     LDA #$49:STA OSC3AD
206 ↑     LDA #$69:STA OSC3SR
207 ↑     LDA #$00:STA FILMOD
210 LDA #<SZOLAM1:STA POINTER1
211 LDA #>SZOLAM1:STA POINTER1+1
212 LDA #<SZOLAM2:STA POINTER2
213 LDA #>SZOLAM2:STA POINTER2+1
214 LDA #<SZOLAM3:STA POINTER3
215 LDA #>SZOLAM3:STA POINTER3+1
220 LDA #0:STA COUNTER1:STA FLAG1
221 ↑        STA COUNTER2:STA FLAG2
222 ↑        STA COUNTER3:STA FLAG3
230 LDA # 16:STA WAVE1
231 LDA # 16:STA WAVE2
232 LDA # 32:STA WAVE3
240 LDA #120:STA SPEED
250 LDA #  0:STA FLAGX
500 LOOP1 INC COUNTER1
501 LDA COUNTER1:LDY #0:CMP (POINTER1),Y
502 BCC GATE1
504 INC POINTER1:BNE *+4:INC POINTER1+1
505 LDY #0:LDA (POINTER1),Y:STA OSC1LF
506 INC POINTER1:BNE *+4:INC POINTER1+1
507 LDY #0:LDA (POINTER1),Y:STA OSC1HF
508 INC POINTER1:BNE *+4:INC POINTER1+1
510 LDA #0:STA COUNTER1
511 LDA WAVE1:ORA #1:STA OSC1WV
512 JMP LOOP2
520 GATE1 LDA COUNTER1:ASL
521 ↑      LDY #0:CMP (POINTER1),Y
522 ↑      BCC LOOP2
523 ↑      LDA WAVE1:STA OSC1WV
600 LOOP2 INC COUNTER2
601 LDA COUNTER2:LDY #0:CMP (POINTER2),Y
602 BCC GATE2
604 INC POINTER2:BNE *+4:INC POINTER2+1
605 LDY #0:LDA (POINTER2),Y:STA OSC2LF
606 INC POINTER2:BNE *+4:INC POINTER2+1
607 LDY #0:LDA (POINTER2),Y:STA OSC2HF
608 INC POINTER2:BNE *+4:INC POINTER2+1
```

```
610 LDA #0:STA COUNTER2
611 LDA WAVE2:ORA #1:STA OSC2WV
612 JMP LOOP3
620 GATE2 LDA COUNTER2:ASL
621 ↑      LDY #0:CMP (POINTER2),Y
622 ↑      BCC LOOP3
623 ↑      LDA WAVE2:STA OSC2WV
700 LOOP3 INC COUNTER3
701 LDA COUNTER3:LDY #0:CMP (POINTER3),Y
702 BCC GATE3
704 INC POINTER3:BNE *+4:INC POINTER3+1
705 LDY #0:LDA (POINTER3),Y:STA OSC3LF
706 INC POINTER3:BNE *+4:INC POINTER2+1
707 LDY #0:LDA (POINTER3),Y:STA OSC3HF
708 INC POINTER3:BNE *+4:INC POINTER3+1
710 LDA #0:STA COUNTER3
711 LDA WAVE3:ORA #1:STA OSC3WV
712 JMP FLAG
720 GATE3 LDA COUNTER3:ASL
721 ↑      LDY #0:CMP (POINTER3),Y
722 ↑      BCC FLAG
723 ↑      LDA WAVE3:STA OSC3WV
800 FLAG *=*
802 LDA POINTER1  :CMP #<POS11-1:BNE N1
803 LDA POINTER1+1:CMP #>POS11-1:BNE N1
804 LDY #0 :LDA (POINTER1),Y
805 TAX:DEX:CPX COUNTER1:BNE N1
806 INC FLAG1
807 LDA FLAG1:CMP #4:BEQ N1
810 LDA #<SZOLAM1:STA POINTER1
811 LDA #>SZOLAM1:STA POINTER1+1
820 N1   *=*
822 LDA POINTER2  :CMP #<POS21-1:BNE N2
823 LDA POINTER2+1:CMP #>POS21-1:BNE N2
824 LDY #0 :LDA (POINTER2),Y
825 TAX:DEX:CPX COUNTER2:BNE N2
826 INC FLAG2
827 LDA FLAG2:CMP #4:BEQ N2
830 LDA #<SZOLAM2:STA POINTER2
831 LDA #>SZOLAM2:STA POINTER2+1
840 N2   *=*
841 LDA POINTER2  :CMP #<POS22-1:BNE N3
842 LDA POINTER2+1:CMP #>POS22-1:BNE N3
843 LDY #0 :LDA (POINTER2),Y
844 TAX:DEX:CPX COUNTER2:BNE N3
845 LDA #180:STA SPEED
850 N3   *=*
851 LDA POINTER1  :CMP #<POS12-1:BNE N4
852 LDA POINTER1+1:CMP #>POS12-1:BNE N4
853 LDY #0 :LDA (POINTER1),Y
854 TAX:DEX:CPX COUNTER1:BNE N4
855 LDA #0:STA FLAG1
856 LDA #1:STA FLAGX
857 LDA #<SZOLAM1:STA POINTER1
858 LDA #>SZOLAM1:STA POINTER1+1
859 LDA #120:STA SPEED
860 N4   *=*
```

```
861 LDA POINTER2  :CMP #<POS23-1:BNE N5
862 LDA POINTER2+1:CMP #>POS23-1:BNE N5
863 LDY #0 :LDA (POINTER2),Y
864 TAX:DEX:CPX COUNTER2:BNE N5
865 LDA #0:STA FLAG2
866 LDA #1:STA FLAGX
867 LDA #<SZOLAM2:STA POINTER2
868 LDA #>SZOLAM2:STA POINTER2+1
870 N5   *=*
871 LDA POINTER1  :CMP #<POS13-1:BNE N6
872 LDA POINTER1+1:CMP #>POS13-1:BNE N6
873 LDY #0 :LDA (POINTER1),Y
874 TAX:DEX:CPX COUNTER1:BNE N6
875 LDA FLAGX:BEQ N6
876 LDA #<NEW1:STA POINTER1
877 LDA #>NEW1:STA POINTER1+1
878 LDA #60:STA SPEED
880 N6   *=*
881 LDA POINTER2  :CMP #<POS24-1:BNE N7
882 LDA POINTER2+1:CMP #>POS24-1:BNE N7
883 LDY #0 :LDA (POINTER2),Y
884 TAX:DEX:CPX COUNTER2:BNE N7
885 LDA FLAGX:BEQ N7
886 LDA #<NEW2:STA POINTER2
887 LDA #>NEW2:STA POINTER2+1
890 ↑     LDA #$25:STA OSC1AD
891 ↑     LDA #$7A:STA OSC1SR
892 ↑     LDA #$38:STA OSC2AD
893 ↑     LDA #$98:STA OSC2SR
894 ↑     LDA #$D0:STA FILTHF
895 ↑     LDA #$03:STA FILMOD
896 ↑     LDA #$06:STA OSC1PH
897 ↑     LDA #$04:STA OSC2PH
898 ↑     LDA #$40:STA WAVE1
899 ↑              STA WAVE2
900 N7   *=*
902 LDA POINTER1  :CMP #<POS15-1:BNE N8
903 LDA POINTER1+1:CMP #>POS15-1:BNE N8
904 LDY #0 :LDA (POINTER1),Y
905 TAX:DEX:CPX COUNTER1:BNE N8
906 LDA #<POS14-1:STA POINTER1
907 LDA #>POS14-1:STA POINTER1+1
908 LDA #100:STA COUNTER1
910 N8   *=*
912 LDA POINTER2  :CMP #<POS26-1:BNE N9
913 LDA POINTER2+1:CMP #>POS26-1:BNE N9
914 LDY #0 :LDA (POINTER2),Y
915 TAX:DEX:CPX COUNTER2:BNE N9
916 LDA #<POS25-1:STA POINTER2
917 LDA #>POS25-1:STA POINTER2+1
918 LDA #100:STA COUNTER2
920 N9   *=*
990 WAIT  LDX SPEED:LDY #0
991 LOOP  DEY:BNE LOOP:DEX:BNE LOOP
992 JMP LOOP1
1000 SZOLAM1 .BYTE 1
1001 .WORD   0:.BYTE 2
```

```
1002 .WORD FS4:.BYTE 2
1003 .WORD CS5:.BYTE 2
1004 .WORD  D5:.BYTE 2
1005 .WORD   0:.BYTE 2
1006 .WORD FS4:.BYTE 2
1007 .WORD CS5:.BYTE 2
1008 .WORD  D5:.BYTE 2
1009 .WORD   0:.BYTE 2
1010 .WORD FS4:.BYTE 2
1011 .WORD CS5:.BYTE 2
1012 .WORD  D5:.BYTE 2
1013 .WORD CS5:.BYTE 2
1014 .WORD  D5:.BYTE 2
1015 .WORD CS5:.BYTE 2
1016 .WORD  D5:.BYTE 2
1017 .WORD   0:.BYTE 2
1018 .WORD  G4:.BYTE 2
1019 .WORD CS5:.BYTE 2
1020 .WORD  D5:.BYTE 2
1021 .WORD   0:.BYTE 2
1022 .WORD  G4:.BYTE 2
1023 .WORD CS5:.BYTE 2
1024 .WORD  D5:.BYTE 2
1025 .WORD   0:.BYTE 2
1026 .WORD  G4:.BYTE 2
1027 .WORD CS5:.BYTE 2
1028 .WORD  D5:.BYTE 2
1029 .WORD CS5:.BYTE 2
1030 .WORD  D5:.BYTE 2
1031 .WORD CS5:.BYTE 2
1032 .WORD  D5:.BYTE 2
1033 POS11 *=*
1034 .WORD  A5:.BYTE 2
1035 .WORD  F5:.BYTE 2
1036 .WORD  E5:.BYTE 2
1037 .WORD  F5:.BYTE 2
1038 .WORD  D5:.BYTE 2
1039 .WORD  F5:.BYTE 2
1040 .WORD  E5:.BYTE 2
1041 .WORD  F5:.BYTE 2
1042 .WORD  A5:.BYTE 2
1043 .WORD  F5:.BYTE 2
1044 .WORD  E5:.BYTE 2
1045 .WORD  F5:.BYTE 2
1046 .WORD  E5:.BYTE 2
1047 .WORD  F5:.BYTE 2
1048 .WORD  E5:.BYTE 2
1049 .WORD  F5:.BYTE 2
1050 .WORD  B5:.BYTE 2
1051 .WORD  F5:.BYTE 2
1052 .WORD  E5:.BYTE 2
1053 .WORD  F5:.BYTE 2
1054 .WORD  D5:.BYTE 2
1055 .WORD  F5:.BYTE 2
1056 .WORD  E5:.BYTE 2
1057 .WORD  F5:.BYTE 2
1058 .WORD  B5:.BYTE 2
1059 .WORD  F5:.BYTE 2
1060 .WORD  E5:.BYTE 2
1061 .WORD  F5:.BYTE 2
1062 .WORD  D5:.BYTE 2
1063 .WORD  F5:.BYTE 2
1064 .WORD  E5:.BYTE 2
1065 .WORD  D5:.BYTE 2
1066 .WORD   0:.BYTE 2
1067 .WORD  F4:.BYTE 2
1068 .WORD  D5:.BYTE 2
1069 .WORD  B4:.BYTE 2
1070 .WORD   0:.BYTE 2
1071 .WORD  G4:.BYTE 2
1072 .WORD  E5:.BYTE 2
1073 .WORD  C5:.BYTE 2
1074 POS13 *=*
1075 .WORD  G5:.BYTE 2
1076 .WORD  E5:.BYTE 2
1077 .WORD  C5:.BYTE 2
1078 .WORD  A4:.BYTE 2
1079 .WORD  F4:.BYTE 2
1080 .WORD  C5:.BYTE 2
1081 .WORD  A4:.BYTE 2
1082 POS12 *=*
1083 NEW1 .BYTE 1
1084 .WORD  G2:.BYTE 16
1085 .WORD  G2:.BYTE 16
1086 .WORD  G2:.BYTE 16
1087 .WORD  G2:.BYTE 16
1088 .WORD  G2:.BYTE 16
1089 .WORD  G2:.BYTE 16
1090 .WORD  G2:.BYTE 16
1091 .WORD  G2:.BYTE 16
1092 .WORD  G2:.BYTE 16
1093 .WORD  G2:.BYTE 16
1094 .WORD  G2:.BYTE 16
1095 .WORD  G2:.BYTE 16
1096 .WORD  G2:.BYTE 16
1097 .WORD  G2:.BYTE 16
1098 .WORD  G2:.BYTE 16
1099 .WORD  G2:.BYTE 16
1100 .WORD  F2:.BYTE 16
1101 .WORD  F2:.BYTE 16
1102 .WORD  E2:.BYTE 16
1103 .WORD  E2:.BYTE 16
1104 .WORD  D2:.BYTE 16
1105 .WORD  C2:.BYTE 16
1106 .WORD  C2:.BYTE 8
1107 .WORD  H1:.BYTE 8
1108 .WORD CS2:.BYTE 32
1109 .WORD CS3:.BYTE 8
1110 .WORD  H2:.BYTE 8
1111 .WORD  A2:.BYTE 16
1112 .WORD  A2:.BYTE 16
1113 .WORD  A2:.BYTE 16
1114 .WORD  A2:.BYTE 6
1115 .WORD  G2:.BYTE 18
```

```
1116 .WORD FS2:.BYTE 8
1117 POS14 *=*
1118 .WORD  0:.BYTE 2
1119 .WORD D4:.BYTE 2
1120 .WORD F4:.BYTE 6
1121 .WORD D4:.BYTE 2
1122 .WORD F4:.BYTE 2
1123 .WORD D4:.BYTE 2
1124 .WORD F4:.BYTE 2
1125 .WORD D4:.BYTE 2
1126 .WORD F4:.BYTE 2
1127 .WORD D4:.BYTE 2
1128 .WORD F4:.BYTE 2
1129 .WORD D4:.BYTE 2
1130 .WORD F4:.BYTE 2
1131 .WORD D4:.BYTE 2
1132 .WORD  0:.BYTE 2
1133 .WORD D4:.BYTE 2
1134 .WORD F4:.BYTE 6
1135 .WORD D4:.BYTE 2
1136 .WORD F4:.BYTE 2
1137 .WORD D4:.BYTE 2
1138 .WORD F4:.BYTE 2
1139 .WORD D4:.BYTE 2
1140 .WORD F4:.BYTE 2
1141 .WORD D4:.BYTE 2
1142 .WORD F4:.BYTE 2
1143 .WORD D4:.BYTE 2
1144 .WORD F4:.BYTE 2
1145 .WORD D4:.BYTE 2
1146 .WORD  0:.BYTE 2
1147 .WORD D4:.BYTE 2
1148 .WORD F4:.BYTE 6
1149 .WORD D4:.BYTE 2
1150 .WORD F4:.BYTE 2
1151 .WORD D4:.BYTE 2
1152 .WORD F4:.BYTE 2
1153 .WORD D4:.BYTE 2
1154 .WORD F4:.BYTE 2
1155 .WORD D4:.BYTE 2
1156 .WORD F4:.BYTE 2
1157 .WORD D4:.BYTE 2
1158 .WORD F4:.BYTE 2
1159 .WORD D4:.BYTE 2
1160 .WORD  0:.BYTE 2
1161 .WORD D4:.BYTE 2
1162 .WORD F4:.BYTE 6
1163 .WORD D4:.BYTE 2
1164 .WORD F4:.BYTE 2
1165 .WORD D4:.BYTE 2
1166 .WORD F4:.BYTE 2
1167 .WORD D4:.BYTE 2
1168 .WORD F4:.BYTE 2
1169 .WORD D4:.BYTE 2
1170 .WORD F4:.BYTE 2
1171 .WORD D4:.BYTE 2
1172 .WORD F4:.BYTE 2
1173 .WORD D4:.BYTE 2
1174 .WORD  0:.BYTE 2
1175 .WORD A4:.BYTE 4
1176 .WORD C5:.BYTE 4
1177 .WORD A4:.BYTE 2
1178 .WORD C5:.BYTE 2
1179 .WORD A4:.BYTE 2
1180 .WORD C5:.BYTE 2
1181 .WORD A4:.BYTE 2
1182 .WORD C5:.BYTE 2
1183 .WORD A4:.BYTE 2
1184 .WORD C5:.BYTE 2
1185 .WORD A4:.BYTE 2
1186 .WORD C5:.BYTE 2
1187 .WORD A4:.BYTE 2
1188 .WORD  0:.BYTE 2
1189 .WORD A4:.BYTE 4
1190 .WORD C5:.BYTE 4
1191 .WORD A4:.BYTE 2
1192 .WORD C5:.BYTE 2
1193 .WORD A4:.BYTE 2
1194 .WORD C5:.BYTE 2
1195 .WORD A4:.BYTE 2
1196 .WORD C5:.BYTE 2
1197 .WORD A4:.BYTE 2
1198 .WORD C5:.BYTE 2
1199 .WORD A4:.BYTE 2
1200 .WORD C5:.BYTE 2
1201 .WORD A4:.BYTE 2
1202 .WORD  0:.BYTE 2
1203 .WORD A4:.BYTE 4
1204 .WORD C5:.BYTE 4
1205 .WORD A4:.BYTE 2
1206 .WORD C5:.BYTE 2
1207 .WORD A4:.BYTE 2
1208 .WORD C5:.BYTE 2
1209 .WORD A4:.BYTE 2
1210 .WORD C5:.BYTE 2
1211 .WORD A4:.BYTE 2
1212 .WORD C5:.BYTE 2
1213 .WORD A4:.BYTE 2
1214 .WORD C5:.BYTE 2
1215 .WORD A4:.BYTE 2
1216 .WORD  0:.BYTE 2
1217 .WORD A4:.BYTE 4
1218 .WORD C5:.BYTE 4
1219 .WORD A4:.BYTE 2
1220 .WORD C5:.BYTE 2
1221 .WORD A4:.BYTE 2
1222 .WORD C5:.BYTE 2
1223 .WORD A4:.BYTE 2
1224 .WORD C5:.BYTE 2
1225 .WORD A4:.BYTE 2
1226 .WORD C5:.BYTE 2
1227 .WORD A4:.BYTE 2
1228 .WORD C5:.BYTE 2
1229 .WORD A4:.BYTE 2
```

```
1230 POS15 *=*
2000 SZOLAM2 .BYTE 1
2001 .WORD  A3:.BYTE 4
2002 .WORD  D4:.BYTE 4
2003 .WORD  A3:.BYTE 4
2004 .WORD  D4:.BYTE 4
2005 .WORD  A3:.BYTE 4
2006 .WORD  D4:.BYTE 4
2007 .WORD  D4:.BYTE 4
2008 .WORD  D4:.BYTE 4
2009 .WORD  A3:.BYTE 4
2010 .WORD  E4:.BYTE 4
2011 .WORD  A3:.BYTE 4
2012 .WORD  E4:.BYTE 4
2013 .WORD  A3:.BYTE 4
2014 .WORD  E4:.BYTE 4
2015 .WORD  E4:.BYTE 4
2016 .WORD  E4:.BYTE 4
2017 POS21 *=*
2018 .WORD  D3:.BYTE 2
2019 .WORD  A3:.BYTE 2
2020 .WORD  D4:.BYTE 2
2021 .WORD  A3:.BYTE 2
2022 .WORD  D3:.BYTE 2
2023 .WORD  A3:.BYTE 2
2024 .WORD  E4:.BYTE 2
2025 .WORD  D4:.BYTE 2
2026 .WORD  D3:.BYTE 2
2027 .WORD  A3:.BYTE 2
2028 .WORD  D4:.BYTE 2
2029 .WORD  A3:.BYTE 2
2030 .WORD  D3:.BYTE 2
2031 .WORD  A3:.BYTE 2
2032 .WORD  E4:.BYTE 2
2033 .WORD  D4:.BYTE 2
2034 .WORD  D3:.BYTE 2
2035 .WORD  B3:.BYTE 2
2036 .WORD  F4:.BYTE 2
2037 .WORD  D4:.BYTE 2
2038 .WORD  D3:.BYTE 2
2039 .WORD  B3:.BYTE 2
2040 .WORD  F4:.BYTE 2
2041 .WORD  D4:.BYTE 2
2042 .WORD  D3:.BYTE 2
2043 .WORD  B4:.BYTE 2
2044 .WORD  F4:.BYTE 2
2045 .WORD  D4:.BYTE 2
2046 .WORD  D3:.BYTE 2
2047 .WORD  B3:.BYTE 2
2048 .WORD  F4:.BYTE 2
2049 .WORD  D4:.BYTE 2
2050 POS22 *=*
2051 .WORD  D3:.BYTE 2
2052 .WORD  B3:.BYTE 2
2053 .WORD  B3:.BYTE 2
2054 .WORD  B3:.BYTE 2
2055 .WORD  D3:.BYTE 2
2056 .WORD  G3:.BYTE 2
2057 .WORD  G3:.BYTE 2
2058 .WORD  G3:.BYTE 2
2059 POS24 *=*
2060 .WORD  D3:.BYTE 16
2061 POS23 *=*
2062 NEW2 .BYTE 1
2063 .WORD  D4:.BYTE 8
2064 .WORD  D4:.BYTE 8
2065 .WORD  D4:.BYTE 8
2066 .WORD  D4:.BYTE 8
2067 .WORD  C4:.BYTE 8
2068 .WORD  C4:.BYTE 8
2069 .WORD  C4:.BYTE 8
2070 .WORD  C4:.BYTE 8
2071 .WORD  D4:.BYTE 8
2072 .WORD  D4:.BYTE 8
2073 .WORD  D4:.BYTE 8
2074 .WORD  D4:.BYTE 8
2075 .WORD  C4:.BYTE 8
2076 .WORD  C4:.BYTE 8
2077 .WORD  C4:.BYTE 8
2078 .WORD  C4:.BYTE 8
2079 .WORD  H3:.BYTE 1
2080 .WORD  C4:.BYTE 1
2081 .WORD  D4:.BYTE 1
2082 .WORD  G4:.BYTE 1
2083 .WORD  H3:.BYTE 1
2084 .WORD  C4:.BYTE 1
2085 .WORD  D4:.BYTE 1
2086 .WORD  G4:.BYTE 1
2087 .WORD  H3:.BYTE 1
2088 .WORD  C4:.BYTE 1
2089 .WORD  D4:.BYTE 1
2090 .WORD  G4:.BYTE 1
2091 .WORD  H3:.BYTE 1
2092 .WORD  C4:.BYTE 1
2093 .WORD  D4:.BYTE 1
2094 .WORD  G4:.BYTE 1
2095 .WORD  H3:.BYTE 1
2096 .WORD  C4:.BYTE 1
2097 .WORD  D4:.BYTE 1
2098 .WORD  G4:.BYTE 1
2099 .WORD  H3:.BYTE 1
2100 .WORD  C4:.BYTE 1
2101 .WORD  D4:.BYTE 1
2102 .WORD  G4:.BYTE 1
2103 .WORD  H3:.BYTE 1
2104 .WORD  C4:.BYTE 1
2105 .WORD  D4:.BYTE 1
2106 .WORD  G4:.BYTE 1
2107 .WORD  H3:.BYTE 1
2108 .WORD  C4:.BYTE 1
2109 .WORD  D4:.BYTE 1
2110 .WORD  G4:.BYTE 1
2111 .WORD  A3:.BYTE 2
2112 .WORD  H3:.BYTE 2
```

```
2113 .WORD  C4:.BYTE 2
2114 .WORD  H3:.BYTE 4
2115 .WORD  A3:.BYTE 2
2116 .WORD  C4:.BYTE 2
2117 .WORD  E4:.BYTE 2
2118 .WORD  A3:.BYTE 2
2119 .WORD  H3:.BYTE 2
2120 .WORD  C4:.BYTE 2
2121 .WORD  H3:.BYTE 4
2122 .WORD  A3:.BYTE 2
2123 .WORD  C4:.BYTE 2
2124 .WORD  E4:.BYTE 2
2125 .WORD  D4:.BYTE 8
2126 .WORD  D4:.BYTE 8
2127 .WORD  D4:.BYTE 8
2128 .WORD  D4:.BYTE 8
2129 .WORD  C4:.BYTE 8
2130 .WORD  C4:.BYTE 8
2131 .WORD  C4:.BYTE 8
2132 .WORD  C4:.BYTE 8
2133 .WORD  A3:.BYTE 8
2134 .WORD  A3:.BYTE 8
2135 .WORD  A3:.BYTE 8
2136 .WORD  A3:.BYTE 8
2137 .WORD  G3:.BYTE 8
2138 .WORD  G3:.BYTE 8
2139 .WORD  G3:.BYTE 8
2140 .WORD FS3:.BYTE 8
2141 .WORD  E3:.BYTE 8
2142 .WORD  E3:.BYTE 8
2143 .WORD  E3:.BYTE 8
2144 .WORD DS3:.BYTE 8
2145 .WORD  E3:.BYTE 8
2146 .WORD  E3:.BYTE 8
2147 .WORD  E3:.BYTE 8
2148 .WORD  E3:.BYTE 8
2149 .WORD  E3:.BYTE 8
2150 .WORD DS3:.BYTE 8
2151 .WORD CS3:.BYTE 8
2152 .WORD CS3:.BYTE 8
2153 .WORD CS3:.BYTE 8
2154 .WORD CS3:.BYTE 8
2155 .WORD CS3:.BYTE 8
2156 .WORD CS3:.BYTE 8
2157 .WORD CS3:.BYTE 6
2158 .WORD CS3:.BYTE 10
2159 .WORD  C3:.BYTE 8
2160 .WORD DS3:.BYTE 8
2161 POS25 *=*
2162 .WORD  B4:.BYTE 6
2163 .WORD  C5:.BYTE 6
2164 .WORD  B4:.BYTE 6
2165 .WORD  C5:.BYTE 6
2166 .WORD  B4:.BYTE 4
2167 .WORD  C5:.BYTE 4
2168 .WORD  B4:.BYTE 6
2169 .WORD  C5:.BYTE 6
2170 .WORD  B4:.BYTE 6
2171 .WORD  C5:.BYTE 6
2172 .WORD  B4:.BYTE 4
2173 .WORD  C5:.BYTE 4
2174 .WORD  B4:.BYTE 6
2175 .WORD  C5:.BYTE 6
2176 .WORD  B4:.BYTE 6
2177 .WORD  C5:.BYTE 6
2178 .WORD  B4:.BYTE 4
2179 .WORD  C5:.BYTE 4
2180 .WORD  B4:.BYTE 6
2181 .WORD  C5:.BYTE 6
2182 .WORD  B4:.BYTE 6
2183 .WORD  C5:.BYTE 6
2184 .WORD  B4:.BYTE 4
2185 .WORD  C5:.BYTE 4
2186 .WORD  F5:.BYTE 6
2187 .WORD  G5:.BYTE 6
2188 .WORD  F5:.BYTE 6
2189 .WORD  E5:.BYTE 6
2190 .WORD  F5:.BYTE 4
2191 .WORD  E5:.BYTE 4
2192 .WORD  F5:.BYTE 6
2193 .WORD  G5:.BYTE 6
2194 .WORD  F5:.BYTE 6
2195 .WORD  E5:.BYTE 6
2196 .WORD  F5:.BYTE 4
2197 .WORD  E5:.BYTE 4
2198 .WORD  F5:.BYTE 6
2199 .WORD  G5:.BYTE 6
2200 .WORD  F5:.BYTE 6
2201 .WORD  E5:.BYTE 6
2202 .WORD  F5:.BYTE 4
2203 .WORD  E5:.BYTE 4
2204 .WORD  F5:.BYTE 6
2205 .WORD  G5:.BYTE 6
2206 .WORD  F5:.BYTE 6
2207 .WORD  E5:.BYTE 6
2208 .WORD  F5:.BYTE 4
2209 .WORD  E5:.BYTE 4
2210 POS26 *=*
3000 SZOLAM3 .BYTE 1
3001 .WORD   0:.BYTE 128
3002 .WORD  D4:.BYTE 6
3003 .WORD FS4:.BYTE 2
3004 .WORD FS4:.BYTE 6
3005 .WORD  D5:.BYTE 2
3006 .WORD  D5:.BYTE 6
3007 .WORD  A4:.BYTE 2
3008 .WORD  A4:.BYTE 8
3009 .WORD  A4:.BYTE 6
3010 .WORD  G4:.BYTE 2
3011 .WORD FS4:.BYTE 4
3012 .WORD  E4:.BYTE 4
3013 .WORD  E4:.BYTE 8
3014 .WORD   0:.BYTE 6
3015 .WORD  E4:.BYTE 2
```

```
3016 .WORD  D4:.BYTE 4
3017 .WORD FS4:.BYTE 4
3018 .WORD FS4:.BYTE 6
3019 .WORD  D5:.BYTE 2
3020 .WORD  D5:.BYTE 4
3021 .WORD  A4:.BYTE 4
3022 .WORD  A4:.BYTE 6
3023 .WORD  A4:.BYTE 2
3024 .WORD  A4:.BYTE 2
3025 .WORD  G4:.BYTE 6
3026 .WORD   0:.BYTE 4
3027 .WORD FS4:.BYTE 2
3028 .WORD  E4:.BYTE 2
3029 .WORD  E4:.BYTE 8
3030 .WORD   0:.BYTE 8
3031 .WORD  D4:.BYTE 12
3032 .WORD  E4:.BYTE 2
3033 .WORD  F4:.BYTE 4
3034 .WORD  A4:.BYTE 4
3035 .WORD  D4:.BYTE 8
3036 .WORD  A4:.BYTE 2
3037 .WORD  G4:.BYTE 10
3038 .WORD  A4:.BYTE 2
3039 .WORD  G4:.BYTE 3
3040 .WORD  A4:.BYTE 1
3041 .WORD  B4:.BYTE 16
3042 .WORD   0:.BYTE 2
3043 .WORD  B4:.BYTE 2
3044 .WORD  C5:.BYTE 2
3045 .WORD  D5:.BYTE 2
3046 .WORD   0:.BYTE 2
3047 .WORD  E5:.BYTE 2
3048 .WORD  F5:.BYTE 2
3049 .WORD  G5:.BYTE 2
3050 .WORD  G5:.BYTE 2
3051 .WORD  A5:.BYTE 2
3052 .WORD  A5:.BYTE 4
3054 .WORD  A5:.BYTE 8
3055 .WORD FS5:.BYTE 4
3056 .WORD  A5:.BYTE 4
3057 .WORD  A5:.BYTE 8
3058 .WORD  A5:.BYTE 16
3059 .WORD   0:.BYTE 96
3060 .WORD  D4:.BYTE 2
3061 .WORD  D4:.BYTE 2
3062 .WORD  D4:.BYTE 4
3063 .WORD   0:.BYTE 4
3064 .WORD FS4:.BYTE 2
3065 .WORD  D5:.BYTE 2
3066 .WORD  D5:.BYTE 4
3067 .WORD  A4:.BYTE 2
3068 .WORD  A4:.BYTE 8
3069 .WORD  A4:.BYTE 2
3070 .WORD  A4:.BYTE 4
3071 .WORD  G4:.BYTE 2
3072 .WORD FS4:.BYTE 4
3073 .WORD  E4:.BYTE 2
3074 .WORD  E4:.BYTE 8
3075 .WORD   0:.BYTE 10
3076 .WORD  D4:.BYTE 2
3077 .WORD  D4:.BYTE 4
3078 .WORD FS4:.BYTE 2
3079 .WORD FS4:.BYTE 6
3080 .WORD   0:.BYTE 2
3081 .WORD  D5:.BYTE 2
3082 .WORD  D5:.BYTE 4
3083 .WORD  A4:.BYTE 2
3084 .WORD  A4:.BYTE 6
3085 .WORD   0:.BYTE 2
3086 .WORD  A4:.BYTE 2
3087 .WORD  A4:.BYTE 2
3088 .WORD  G4:.BYTE 6
3089 .WORD   0:.BYTE 4
3090 .WORD FS4:.BYTE 2
3091 .WORD  G4:.BYTE 2
3092 .WORD  E4:.BYTE 8
3093 .WORD   0:.BYTE 8
3094 .WORD  D4:.BYTE 8
3095 .WORD   0:.BYTE 4
3096 .WORD  E4:.BYTE 2
3097 .WORD  F4:.BYTE 4
3098 .WORD  A4:.BYTE 4
3099 .WORD  D4:.BYTE 6
3100 .WORD   0:.BYTE 2
3101 .WORD  A4:.BYTE 2
3102 .WORD  G4:.BYTE 3
3103 .WORD  G4:.BYTE 1
3104 .WORD  G4:.BYTE 8
3105 .WORD   0:.BYTE 2
3106 .WORD  A4:.BYTE 2
3107 .WORD  B4:.BYTE 8
3108 .WORD   0:.BYTE 8
3109 .WORD   0:.BYTE 2
3110 .WORD  B4:.BYTE 2
3111 .WORD  C5:.BYTE 2
3112 .WORD  D5:.BYTE 2
3113 .WORD   0:.BYTE 2
3114 .WORD  E5:.BYTE 2
3115 .WORD  F5:.BYTE 2
3116 .WORD  G5:.BYTE 2
3117 .WORD  G5:.BYTE 4
3118 .WORD  A5:.BYTE 8
3119 .WORD  G5:.BYTE 2
3120 .WORD  H5:.BYTE 10
3121 .WORD   0:.BYTE 8
3122 .WORD  A5:.BYTE 4
3123 .WORD  A5:.BYTE 4
3124 .WORD   0:.BYTE 20
3125 .WORD  E5:.BYTE 4
3126 .WORD  D5:.BYTE 2
3127 .WORD  D5:.BYTE 2
3128 .WORD  D5:.BYTE 2
3129 .WORD  D5:.BYTE 2
3130 .WORD  D5:.BYTE 4
```

```
3131 .WORD  D5:.BYTE 4
3132 .WORD   0:.BYTE 12
3133 .WORD  H4:.BYTE 4
3134 .WORD  A4:.BYTE 4
3135 .WORD  H4:.BYTE 4
3136 .WORD  C4:.BYTE 8
3137 .WORD  H4:.BYTE 4
3138 .WORD  A4:.BYTE 2
3139 .WORD  A4:.BYTE 6
3140 .WORD   0:.BYTE 68
3141 .WORD  D5:.BYTE 2
3142 .WORD  D5:.BYTE 2
3143 .WORD  D5:.BYTE 2
3144 .WORD  D5:.BYTE 2
3145 .WORD  D5:.BYTE 6
3146 .WORD  D5:.BYTE 2
3147 .WORD  D5:.BYTE 8
3148 .WORD   0:.BYTE 4
3149 .WORD  G4:.BYTE 4
3150 .WORD  A4:.BYTE 6
3151 .WORD  H4:.BYTE 4
3152 .WORD  C5:.BYTE 6
3153 .WORD  H4:.BYTE 6
3154 .WORD  A4:.BYTE 10
3155 .WORD   0:.BYTE 6
3156 .WORD  G4:.BYTE 2
3157 .WORD  H4:.BYTE 4
3158 .WORD  A4:.BYTE 4
3159 .WORD  D5:.BYTE 4
3160 .WORD  C5:.BYTE 2
3161 .WORD  H4:.BYTE 6
3162 .WORD  A4:.BYTE 2
3163 .WORD  H4:.BYTE 10
3164 .WORD   0:.BYTE 12
3165 .WORD  G4:.BYTE 2
3166 .WORD  G4:.BYTE 2
3167 .WORD  A4:.BYTE 2
3168 .WORD  H4:.BYTE 4
3169 .WORD  H4:.BYTE 4
3170 .WORD  C5:.BYTE 4
3171 .WORD  G5:.BYTE 4
3172 .WORD  G5:.BYTE 6
3173 .WORD   0:.BYTE 8
3174 .WORD  H4:.BYTE 6
3175 .WORD  H4:.BYTE 2
3176 .WORD  E5:.BYTE 12
3177 .WORD DS5:.BYTE 2
3178 .WORD DS5:.BYTE 2
3179 .WORD DS5:.BYTE 4
3180 .WORD CS5:.BYTE 8
3181 .WORD CS5:.BYTE 12
3182 .WORD  H4:.BYTE 6
3183 .WORD  A4:.BYTE 2
3184 .WORD  A4:.BYTE 4
3185 .WORD  A4:.BYTE 4
3186 .WORD   0:.BYTE 4
3187 .WORD DS5:.BYTE 2
3188 .WORD DS5:.BYTE 2
3189 .WORD DS5:.BYTE 4
3190 .WORD FS5:.BYTE 4
3191 .WORD  H4:.BYTE 8
3192 .WORD   0:.BYTE 10
3193 .WORD  A4:.BYTE 4
3194 .WORD  A4:.BYTE 2
3195 .WORD  A4:.BYTE 4
3196 .WORD  A4:.BYTE 2
3197 .WORD  G4:.BYTE 6
3198 .WORD  G4:.BYTE 4
3199 .WORD  G4:.BYTE 8
3200 .WORD  A4:.BYTE 8
3201 .WORD  G4:.BYTE 16
3202 .WORD   0:.BYTE 4
3203 .WORD  B4:.BYTE 4
3204 .WORD  C5:.BYTE 4
3205 .WORD  B4:.BYTE 4
3206 .WORD  G4:.BYTE 32
3207 .WORD   0:.BYTE 64
3208 .WORD  D3:.BYTE 6
3209 .WORD  A3:.BYTE 6
3210 .WORD  C4:.BYTE 6
3211 .WORD  A4:.BYTE 6
3212 .WORD  C4:.BYTE 4
3213 .WORD  A4:.BYTE 4
3214 .WORD  D3:.BYTE 6
3215 .WORD  A3:.BYTE 6
3216 .WORD  C4:.BYTE 6
3217 .WORD  A4:.BYTE 6
3218 .WORD  C4:.BYTE 4
3219 .WORD  A4:.BYTE 4
3220 .WORD  D3:.BYTE 6
3221 .WORD  A3:.BYTE 6
3222 .WORD  C4:.BYTE 6
3223 .WORD  A4:.BYTE 6
3224 .WORD  C4:.BYTE 4
3225 .WORD  A4:.BYTE 4
3226 .WORD  D3:.BYTE 6
3227 .WORD  A3:.BYTE 6
3228 .WORD  C4:.BYTE 6
3229 .WORD  A4:.BYTE 6
3230 .WORD  C4:.BYTE 4
3231 .WORD  A4:.BYTE 4
3232 .WORD  G3:.BYTE 32
3233 .WORD  G3:.BYTE 32
3234 .WORD  G3:.BYTE 32
3235 .WORD  G3:.BYTE 32
3236 .WORD  D3:.BYTE 6
3237 .WORD  A3:.BYTE 6
3238 .WORD  C4:.BYTE 6
3239 .WORD  A4:.BYTE 6
3240 .WORD  C4:.BYTE 4
3241 .WORD  A4:.BYTE 4
3242 .WORD  D3:.BYTE 6
3243 .WORD  A3:.BYTE 6
3244 .WORD  C4:.BYTE 6
```

```
3245 .WORD  A4:.BYTE 6
3246 .WORD  C4:.BYTE 4
3247 .WORD  A4:.BYTE 4
3248 .WORD  D3:.BYTE 6
3249 .WORD  A3:.BYTE 6
3250 .WORD  C4:.BYTE 6
3251 .WORD  A4:.BYTE 6
3252 .WORD  C4:.BYTE 4
3253 .WORD  A4:.BYTE 4
3254 .WORD  D3:.BYTE 2
3999 .WORD 0,0,0,0
READY.
```

```
0 REM **** P27. ****
1 :
2 :
1000 POKE 56,128:CLR:FOR I=32768 TO 35690
1010 READ X:POKE I,X:S=S+X:NEXT
1020 DATA 160,  0,169, 29,133,  2,169,128,133,  3,177,  2
1030 DATA 208,  3, 76,133,128, 32,210,255,230,  2,208,  2
1040 DATA 230,  3, 76, 10,128,147, 13, 13, 13, 13, 13, 13
1050 DATA  13, 13, 13, 13, 32, 32, 32, 32, 32, 32, 32, 32
1060 DATA  32, 32, 32, 32, 84, 72, 69, 32, 67, 73, 78, 69
1070 DATA  77, 65, 32, 83, 72, 79, 87, 13, 32, 32, 32, 32
1080 DATA  32, 32, 32, 32, 32, 32, 32, 77, 85, 83, 73, 67
1090 DATA  32, 66, 89, 58, 32, 71, 69, 78, 69, 83, 73, 83
1100 DATA  13, 32, 32, 32, 32, 32, 65, 82, 82, 65, 78, 71
1110 DATA  69, 77, 69, 78, 84, 32, 66, 89, 58, 32, 84, 72
1120 DATA  79, 77, 65, 83, 32, 68, 65, 67, 72, 83, 69, 76
1130 DATA   0,120, 32,144,128, 88,169,  0,141, 24,212, 96
1140 DATA 169, 31,141, 24,212,169, 37,141,  5,212,169,123
1150 DATA 141,  6,212,169, 37,141, 12,212,169,123,141, 13
1160 DATA 212,169, 73,141, 19,212,169,105,141, 20,212,169
1170 DATA   0,141, 23,212,169,102,133,  2,169,131,133,  3
1180 DATA 169,  8,133,  5,169,134,133,  6,169,107,133,  8
1190 DATA 169,136,133,  9,169,  0,133,  4,141,208,  2,133
1200 DATA   7,141,209,  2,133, 10,141,210,  2,169, 16,141
1210 DATA 192,  2,169, 16,141,193,  2,169, 32,141,194,  2
1220 DATA 169,120,141,224,  2,169,  0,141,225,  2,230,  4
1230 DATA 165,  4,160,  0,209,  2,144, 47,230,  2,208,  2
1240 DATA 230,  3,160,  0,177,  2,141,  0,212,230,  2,208
1250 DATA   2,230,  3,160,  0,177,  2,141,  1,212,230,  3
1260 DATA 208,  2,230,  3,169,  0,133,  4,173,192,  2,  9
1270 DATA   1,141,  4,212, 76, 66,129,165,  4, 10,160,  0
1280 DATA 209,  2,144,  6,173,192,  2,141,  4,212,230,  7
1290 DATA 165,  7,160,  0,209,  5,144, 47,230,  5,208,  2
1300 DATA 230,  6,160,  0,177,  5,141,  7,212,230,  5,208
1310 DATA   2,230,  6,160,  0,177,  5,141,  8,212,230,  5
1320 DATA 208,  2,230,  6,169,  0,133,  7,173,193,  2,  9
1330 DATA   1,141, 11,212, 76,138,129,165,  7, 10,160,  0
1340 DATA 209,  5,144,  6,173,193,  2,141, 11,212,230, 10
1350 DATA 165, 10,160,  0,209,  8,144, 54,230,  8,208,  2
1360 DATA 230,  9,160,  0,177,  8,141, 14,212,230,  8,208
1370 DATA   2,230,  9,160,  0,177,  8,141, 15,212,230,  8
1380 DATA 208,  2,230,  9,160,  0,177,  8,208,  1, 96,169
1390 DATA   0,133, 10,173,194,  2,  9,  1,141, 18,212, 76
1400 DATA 217,129,165, 10, 10,160,  0,209,  8,144,  6,173
1410 DATA 194,  2,141, 18,212,165,  2,201,198,208, 34,165
1420 DATA   3,201,131,208, 28,160,  0,177,  2,170,202,228
1430 DATA   4,208, 18,238,208,  2,173,208,  2,201,  4,240
1440 DATA   8,169,102,133,  2,169,131,133,  3,165,  5,201
1450 DATA  56,208, 34,165,  6,201,134,208, 28,160,  0,177
1460 DATA   5,170,202,228,  7,208, 18,238,209,  2,173,209
```

```
1470 DATA   2,201,  4,240,  8,169,  8,133,  5,169,134,133
1480 DATA   6,165,  5,201,152,208, 21,165,  6,201,134,208
1490 DATA  15,160,  0,177,  5,170,202,228,  7,208,  5,169
1500 DATA 180,141,224,  2,165,  2,201, 83,208, 39,165,  3
1510 DATA 201,132,208, 33,160,  0,177,  2,170,202,228,  4
1520 DATA 208, 23,169,  0,141,208,  2,169,  1,141,225,  2
1530 DATA 169,102,133,  2,169,131,133,  3,169,120,141,224
1540 DATA   2,165,  5,201,179,208, 34,165,  6,201,134,208
1550 DATA  28,160,  0,177,  5,170,202,228,  7,208, 18,169
1560 DATA   0,141,209,  2,169,  1,141,225,  2,169,  8,133
1570 DATA   5,169,134,133,  6,165,  2,201, 62,208, 34,165
1580 DATA   3,201,132,208, 28,160,  0,177,  2,170,202,228
1590 DATA   4,208, 18,173,225,  2,240, 13,169, 84,133,  2
1600 DATA 169,132,133,  3,169, 60,141,224,  2,165,  5,201
1610 DATA 176,208, 77,165,  6,201,134,208, 71,160,  0,177
1620 DATA   5,170,202,228,  7,208, 61,173,225,  2,240, 56
1630 DATA 169,180,133,  5,169,134,133,  6,169, 37,141,  5
1640 DATA 212,169,122,141,  6,212,169, 56,141, 12,212,169
1650 DATA 152,141, 13,212,169,208,141, 22,212,169,  3,141
1660 DATA  23,212,169,  6,141,  3,212,169,  4,141, 10,212
1670 DATA 169, 64,141,192,  2,141,193,  2,165,  2,201,  7
1680 DATA 208, 28,165,  3,201,134,208, 22,160,  0,177,  2
1690 DATA 170,202,228,  4,208, 12,169,183,133,  2,169,132
1700 DATA 133,  3,169,100,133,  4,165,  5,201,106,208, 28
1710 DATA 165,  6,201,136,208, 22,160,  0,177,  5,170,202
1720 DATA 228,  7,208, 12,169,218,133,  5,169,135,133,  6
1730 DATA 169,100,133,  7,174,224,  2,160,  0,136,208,253
1740 DATA 202,208,250, 76,250,128,  1,  0,  0,  2,181, 23
1750 DATA   2,134, 35,  2,162, 37,  2,  0,  0,  2,181, 23
1760 DATA   2,134, 35,  2,162, 37,  2,  0,  0,  2,181, 23
1770 DATA   2,134, 35,  2,162, 37,  2,134, 35,  2,162, 37
1780 DATA   2,134, 35,  2,162, 37,  2,  0,  0,  2, 30, 25
1790 DATA   2,134, 35,  2,162, 37,  2,  0,  0,  2, 30, 25
1800 DATA   2,134, 35,  2,162, 37,  2,  0,  0,  2, 30, 25
1810 DATA   2,134, 35,  2,162, 37,  2,134, 35,  2,162, 37
1820 DATA   2,134, 35,  2,162, 37,  2, 99, 56,  2,193, 44
1830 DATA   2, 62, 42,  2,193, 44,  2,162, 37,  2,193, 44
1840 DATA   2, 62, 42,  2,193, 44,  2, 99, 56,  2,193, 44
1850 DATA   2, 62, 42,  2,193, 44,  2, 62, 42,  2,193, 44
1860 DATA   2, 62, 42,  2,193, 44,  2,190, 59,  2,193, 44
1870 DATA   2, 62, 42,  2,193, 44,  2,162, 37,  2,193, 44
1880 DATA   2, 62, 42,  2,193, 44,  2,190, 59,  2,193, 44
1890 DATA   2, 62, 42,  2,193, 44,  2,162, 37,  2,193, 44
1900 DATA   2, 62, 42,  2,162, 37,  2,  0,  0,  2, 96, 22
1910 DATA   2,162, 37,  2,223, 29,  2,  0,  0,  2, 30, 25
1920 DATA   2, 62, 42,  2,135, 33,  2, 60, 50,  2, 62, 42
1930 DATA   2,135, 33,  2, 49, 28,  2, 96, 22,  2,135, 33
1940 DATA   2, 49, 28,  4,  1, 71,  6, 16, 71,  6, 16, 71
1950 DATA   6, 16, 71,  6, 16, 71,  6, 16, 71,  6, 16, 71
1960 DATA   6, 16, 71,  6, 16, 71,  6, 16, 71,  6, 16, 71
1970 DATA   6, 16, 71,  6, 16, 71,  6, 16, 71,  6, 16, 71
1980 DATA   6, 16, 71,  6, 16,152,  5, 16,152,  5, 16, 71
1990 DATA   5, 16, 71,  5,  8,180,  4,  8, 48,  4, 16, 48
2000 DATA   4,  8,244,  3,  8,112,  4, 32,225,  6,  8,233
2010 DATA   7,  8, 12,  7, 16, 12,  7, 16, 12,  7, 16, 12
2020 DATA   7,  6, 71,  6, 18,237,  5,  8,  0,  0,  2,209
2030 DATA  18,  2, 96, 22,  6,209, 18,  2, 96, 22,  2,209
```

```
2040 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2050 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2060 DATA  18,  2,  0,  0,  2,209, 18,  2, 96, 22,  6,209
2070 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2080 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2090 DATA  18,  2, 96, 22,  2,209, 18,  2,  0,  0,  2,209
2100 DATA  18,  2, 96, 22,  6,209, 18,  2, 96, 22,  2,209
2110 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2120 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2130 DATA  18,  2,  0,  0,  2,209, 18,  2, 96, 22,  6,209
2140 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2150 DATA  18,  2, 96, 22,  2,209, 18,  2, 96, 22,  2,209
2160 DATA  18,  2, 96, 22,  2,209, 18,  2,  0,  0,  2, 49
2170 DATA  28,  4,135, 33,  4, 49, 28,  2,135, 33,  2, 49
2180 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2190 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2200 DATA  28,  2,  0,  0,  2, 49, 28,  4,135, 33,  4, 49
2210 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2220 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2230 DATA  28,  2,135, 33,  2, 49, 28,  2,  0,  0,  2, 49
2240 DATA  28,  4,135, 33,  4, 49, 28,  2,135, 33,  2, 49
2250 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2260 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2270 DATA  28,  2,  0,  0,  2, 49, 28,  4,135, 33,  4, 49
2280 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2290 DATA  28,  2,135, 33,  2, 49, 28,  2,135, 33,  2, 49
2300 DATA  28,  2,135, 33,  2, 49, 28,  2,  1, 24, 14,  4
2310 DATA 209, 18,  4, 24, 14,  4,209, 18,  4, 24, 14,  4
2320 DATA 209, 18,  4,209, 18,  4,209, 18,  4, 24, 14,  4
2330 DATA  31, 21,  4, 24, 14,  4, 31, 21,  4, 24, 14,  4
2340 DATA  31, 21,  4, 31, 21,  4, 31, 21,  4,104,  9,  2
2350 DATA  24, 14,  2,209, 18,  2, 24, 14,  2,104,  9,  2
2360 DATA  24, 14,  2, 31, 21,  2,209, 18,  2,104,  9,  2
2370 DATA  24, 14,  2,209, 18,  2, 24, 14,  2,104,  9,  2
2380 DATA  24, 14,  2, 31, 21,  2,209, 18,  2,104,  9,  2
2390 DATA 239, 14,  2, 96, 22,  2,209, 18,  2,104,  9,  2
2400 DATA 239, 14,  2, 96, 22,  2,209, 18,  2,104,  9,  2
2410 DATA 223, 29,  2, 96, 22,  2,209, 18,  2,104,  9,  2
2420 DATA 239, 14,  2, 96, 22,  2,209, 18,  2,104,  9,  2
2430 DATA 239, 14,  2,239, 14,  2,239, 14,  2,104,  9,  2
2440 DATA 143, 12,  2,143, 12,  2,143, 12,  2,104,  9, 16
2450 DATA   1,209, 18,  8,209, 18,  8,209, 18,  8,209, 18
2460 DATA   8,195, 16,  8,195, 16,  8,195, 16,  8,195, 16
2470 DATA   8,209, 18,  8,209, 18,  8,209, 18,  8,209, 18
2480 DATA   8,195, 16,  8,195, 16,  8,195, 16,  8,195, 16
2490 DATA   8,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2500 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2510 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2520 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2530 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2540 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2550 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2560 DATA   1,210, 15,  1,195, 16,  1,209, 18,  1, 30, 25
2570 DATA   1, 24, 14,  2,210, 15,  2,195, 16,  2,210, 15
2580 DATA   4, 24, 14,  2,195, 16,  2, 31, 21,  2, 24, 14
2590 DATA   2,210, 15,  2,195, 16,  2,210, 15,  4, 24, 14
2600 DATA   2,195, 16,  2, 31, 21,  2,209, 18,  8,209, 18
```

```
2610 DATA   8,209, 18,  8,209, 18,  8,195, 16,  8,195, 16
2620 DATA   8,195, 16,  8,195, 16,  8, 24, 14,  8, 24, 14
2630 DATA   8, 24, 14,  8, 24, 14,  8,143, 12,  8,143, 12
2640 DATA   8,143, 12,  8,218, 11,  8,143, 10,  8,143, 10
2650 DATA   8,143, 10,  8,247,  9,  8,143, 10,  8,143, 10
2660 DATA   8,143, 10,  8,143, 10,  8,143, 10,  8,247,  9
2670 DATA   8,225,  8,  8,225,  8,  8,225,  8,  8,225,  8
2680 DATA   8,225,  8,  8,225,  8,  8,225,  8,  6,225,  8
2690 DATA  10, 97,  8,  8,247,  9,  8,223, 29,  6,135, 33
2700 DATA   6,223, 29,  6,135, 33,  6,223, 29,  4,135, 33
2710 DATA   4,223, 29,  6,135, 33,  6,223, 29,  6,135, 33
2720 DATA   6,223, 29,  4,135, 33,  4,223, 29,  6,135, 33
2730 DATA   6,223, 29,  6,135, 33,  6,223, 29,  4,135, 33
2740 DATA   4,223, 29,  6,135, 33,  6,223, 29,  6,135, 33
2750 DATA   6,223, 29,  4,135, 33,  4,193, 44,  6, 60, 50
2760 DATA   6,193, 44,  6, 62, 42,  6,193, 44,  4, 62, 42
2770 DATA   4,193, 44,  6, 60, 50,  6,193, 44,  6, 62, 42
2780 DATA   6,193, 44,  4, 62, 42,  4,193, 44,  6, 60, 50
2790 DATA   6,193, 44,  6, 62, 42,  6,193, 44,  4, 62, 42
2800 DATA   4,193, 44,  6, 60, 50,  6,193, 44,  6, 62, 42
2810 DATA   6,193, 44,  4, 62, 42,  4,  1,  0,  0,128,209
2820 DATA  18,  6,181, 23,  2,181, 23,  6,162, 37,  2,162
2830 DATA  37,  6, 49, 28,  2, 49, 28,  8, 49, 28,  6, 30
2840 DATA  25,  2,181, 23,  4, 31, 21,  4, 31, 21,  8,  0
2850 DATA   0,  6, 31, 21,  2,209, 18,  4,181, 23,  4,181
2860 DATA  23,  6,162, 37,  2,162, 37,  4, 49, 28,  4, 49
2870 DATA  28,  6, 49, 28,  2, 49, 28,  2, 30, 25,  6,  0
2880 DATA   0,  4,181, 23,  2, 31, 21,  2, 31, 21,  8,  0
2890 DATA   0,  8,209, 18, 12, 31, 21,  2, 96, 22,  4, 49
2900 DATA  28,  4,209, 18,  8, 49, 28,  2, 30, 25, 10, 49
2910 DATA  28,  2, 30, 25,  3, 49, 28,  1,223, 29, 16,  0
2920 DATA   0,  2,223, 29,  2,135, 33,  2,162, 37,  2,  0
2930 DATA   0,  2, 62, 42,  2,193, 44,  2, 60, 50,  2, 60
2940 DATA  50,  2, 99, 56,  2, 99, 56,  4, 99, 56,  8,107
2950 DATA  47,  4, 99, 56,  4, 99, 56,  8, 99, 56, 16,  0
2960 DATA   0, 96,209, 18,  2,209, 18,  2,209, 18,  4,  0
2970 DATA   0,  4,181, 23,  2,162, 37,  2,162, 37,  4, 49
2980 DATA  28,  2, 49, 28,  8, 49, 28,  2, 49, 28,  4, 30
2990 DATA  25,  2,181, 23,  4, 31, 21,  2, 31, 21,  8,  0
3000 DATA   0, 10,209, 18,  2,209, 18,  4,181, 23,  2,181
3010 DATA  23,  6,  0,  0,  2,162, 37,  2,162, 37,  4, 49
3020 DATA  28,  2, 49, 28,  6,  0,  0,  2, 49, 28,  2, 49
3030 DATA  28,  2, 30, 25,  6,  0,  0,  4,181, 23,  2, 30
3040 DATA  25,  2, 31, 21,  8,  0,  0,  8,209, 18,  8,  0
3050 DATA   0,  4, 31, 21,  2, 96, 22,  4, 49, 28,  4,209
3060 DATA  18,  6,  0,  0,  2, 49, 28,  2, 30, 25,  3, 30
3070 DATA  25,  1, 30, 25,  8,  0,  0,  2, 49, 28,  2,223
3080 DATA  29,  8,  0,  0,  8,  0,  0,  2,223, 29,  2,135
3090 DATA  33,  2,162, 37,  2,  0,  0,  2, 62, 42,  2,193
3100 DATA  44,  2, 60, 50,  2, 60, 50,  4, 99, 56,  8, 60
3110 DATA  50,  2, 75, 63, 10,  0,  0,  8, 99, 56,  4, 99
3120 DATA  56,  4,  0,  0, 20, 62, 42,  4,162, 37,  2,162
3130 DATA  37,  2,162, 37,  2,162, 37,  2,162, 37,  4,162
3140 DATA  37,  4,  0,  0, 12,165, 31,  4, 49, 28,  4,165
3150 DATA  31,  4,135, 33,  8,165, 31,  4, 49, 28,  2, 49
3160 DATA  28,  6,  0,  0, 68,162, 37,  2,162, 37,  2,162
3170 DATA  37,  2,162, 37,  2,162, 37,  6,162, 37,  2,162
```

```
3180 DATA 37,  8,  0,  0,  4, 30, 25,  4, 49, 28,  6,165
3190 DATA 31,  4,135, 33,  6,165, 31,  6, 49, 28, 10,  0
3200 DATA  0,  6, 30, 25,  2,165, 31,  4, 49, 28,  4,162
3210 DATA 37,  4,135, 33,  2,165, 31,  6, 49, 28,  2,165
3220 DATA 31, 10,  0,  0, 12, 30, 25,  2, 30, 25,  2, 49
3230 DATA 28,  2,165, 31,  4,165, 31,  4,135, 33,  4, 30
3240 DATA 25,  4, 30, 25,  6,  0,  0,  8,165, 31,  6,165
3250 DATA 31,  2, 62, 42, 12,223, 39,  2,223, 39,  2,223
3260 DATA 39,  4,134, 35,  8,134, 35, 12,165, 31,  6, 49
3270 DATA 28,  2, 49, 28,  4, 49, 28,  4,  0,  0,  4,223
3280 DATA 39,  2,223, 39,  2,223, 39,  4,107, 47,  4,165
3290 DATA 31,  8,  0,  0, 10, 49, 28,  4, 49, 28,  2, 49
3300 DATA 28,  4, 49, 28,  2, 30, 25,  6, 30, 25,  4, 30
3310 DATA 25,  8, 49, 28,  8, 30, 25, 16,  0,  0,  4,223
3320 DATA 29,  4,135, 33,  4,223, 29,  4, 30, 25, 32,  0
3330 DATA ,0, 64,104,  9,  6, 24, 14,  6,195, 16,  6, 49
3340 DATA 28,  6,195, 16,  4, 49, 28,  4,104,  9,  6, 24
3350 DATA 14,  6,195, 16,  6, 49, 28,  6,195, 16,  4, 49
3360 DATA 28,  4,104,  9,  6, 24, 14,  6,195, 16,  6, 49
3370 DATA 28,  6,195, 16,  4, 49, 28,  4,104,  9,  6, 24
3380 DATA 14,  6,185, 16,  6, 49, 28,  6,195, 16,  4, 49
3390 DATA 28,  4,143, 12, 32,143, 12, 32,143, 12, 32,143
3400 DATA 12, 32,104,  9,  6, 24, 14,  6,195, 16,  6, 49
3410 DATA 28,  6,195, 16,  4, 49, 28,  4,104,  9,  6, 24
3420 DATA 14,  6,195, 16,  6, 49, 28,  6,195, 16,  4, 49
3430 DATA 28,  4,104,  9,  6, 24, 14,  6,195, 16,  6, 49
3440 DATA 28,  6,195, 16,  4, 49, 28,  4,104,  9,  2,  0
3450 DATA  0,  0,  0,  0,  0,  0,  0
3460 IF S<>191135 THEN PRINT"HIBA A DATA-SORBAN..!!":END
3470 PRINT"OK - START: SYS 32768-AL"
READY.
```

5.9. Frekvenciamoduláció

Az előző fejezetben bemutatott hosszú assembler program után foglalkozzunk még néhány rövidebb, a hang chipet programozó assembler alkalmazással

Erre a fejezetre már többször hivatkoztunk a 4. fejezetben. Olyankor tettük ezt, amikor a BASIC a hang-chip programozására alkalmatlannak mutatkozott. Egyik ilyen eset a frekvenciamoduláció.

Az eddigiek során a frekvencia konstans volt a hang hangzásának időtartama alatt. Felmerül a kérdés: mi történik akkor, ha a hangfrekvenciát gyorsan változtatjuk?

Azzal a sebességgel, amelyet az assembler kínál, számtalan hangeffektust lehet programozni. De még mielőtt belemélyednénk a frekvenciamodulációs technikába, nézzük meg, mit is jelent a moduláció.

A negyedik fejezetben a gyűrűsmoduláció hatásáról írtunk. A frekvenciamodulációt is hasonlóképpen képzelhetjük el. Rendelkezésre áll egy bázis oszcillátor és egy vezérlő oszcillátor, amelyik a vezérlő hatást kifejti.
Modulálni csak a bázis oszcillátor meghatározott regisztereit lehet. Ennek a folyamatnak a lényegét a modulálás szó helyett inkább a „folyamatos változtatás" kifejezés tükrözi hűen. Ez azt jelenti, hogy a zene nem egy jól meghatározott A, B és C frekvencián szólal meg, ahol az A, B és C hangok hangzási időtartama a műben meghatározott hosszúságú, hanem a frekvencia folyamatosan változik.

A mintaprogramban az első oszcillátor bázisként, a harmadik pedig vezérlő oszcillátorként működik. Ebben az esetben két olyan regiszterre van szükségünk, melyeket ez ideig csak érintőlegesen említettünk: a $D41B regiszterre és a $D418 regiszter "3 KI" bitjére.

A $D41B regiszter teszi lehetővé, hogy a mikroprocesszor kiolvassa a 3. oszcillátor jelének digitális értékét. Ez az érték kerül be az 1. oszcillátor frekvenciájába. Ahhoz, hogy csak az első oszcillátor legyen hallható, a 3. oszcillátor pedig nem – mivel annak csak moduláló hatásúnak kell lennie –, a "3 KI" bit beállításával a 3. oszcillátort "hangtalanná" tesszük.

A program egyetlen meghatározott hangmagassághoz sem rendelhető, folyamatosan változó frekvenciákat szolgáltat. A 3. oszcillátor frekvenciájától függ, hogy az 1. oszcillátort milyen módon moduláljuk.

A programban a 3. oszcillátor frekvenciáját különféle módszerekkel határozzuk meg. Az első BASIC betöltésnél a számláló értéke $A2, így a 3. oszcillátor frekvenciáját a megszakító program által vezérelt órából (TI$) generáljuk.

A második BASIC betöltésnél a számláló értéke $C5. Így most a billentyűzeten lenyomott jeltől függ, hogy milyen hangot hallunk. Próbáljuk ki a funkcióbillentyűket vagy a RETURN-t!

5.10. A megszakítás és a zeneprogramozás

Az ötödik fejezet végén foglalkozzunk a zenei programozás egy különleges módjával, a megszakító eljárással. Látni fogjuk, hogy ez valójában egyszerű dolog.

Az ún. megszakítási technika található a következő oldalak mintaprogramjaiban. Ha a gépi kódú programot a megfelelő SYS utasítással elindítjuk, láthatjuk, hogy a BASIC azonnal visszajelez READY-vel. Ezután egy tetszés szerinti billentyű lenyomását követően egy hangot hallunk, ami billentyűnként más.
Ez a megszakítás addig funkcionál, amíg a megszakítás vektort felül nem írjuk, pl.: a RUN/STOP és a RESTORE billentyűk egyidejű lenyomásával.

A hangfrekvenciák lenyomott billentyűkből való kiszámításának módszere a két programban eltérő. Az első programnál a 197-es vagy a 162-es cím tartalmát közvetlenül a frekvencia felső byte-jaként használjuk.

A 197-es cím esetén a billentyűzet lekérdezésének indexét, a 162-es cím esetén a megszakítás vezérelte óra (TI$) alsó byte-ját használjuk. Eredményül eléggé hamis hangzást kapunk, ami a hozzá nem értők számára is hatásos lehet, ameddig a megszakítás aktív, addig a BASIC-ben szerkeszteni lehet (pl. lemeztartalom leolvasása, programok kinyomtatása).

A második program lehetővé teszi, hogy zongorabillentyűként használjuk a Commodore 64-es billentyűzetét. Könnyen meg tudjuk találni, hogy melyik billentyűhöz milyen hangmagasság van hozzárendelve.
Kb. 3 fél oktávot fog át a két felső és a két alsó billentyűsor összesen.

A Profi-Ass program bevitele után tetszés szerint módosíthatjuk a billentyűzethez való hozzárendelést. A megszakítás kapcsán jó ötletként vetődik fel olyan BASIC program írása, mely lehetővé teszi a felhasználó számára pl. a hullámforma vagy a burkológörbe módosítását.
Ha valaki nagyon kitartó, komplex zeneprogrammá bővítheti a megszakítást, amely tárolja a hangot, a dallamot stb. A lenyomott billentyűt pl. egy GET ciklussal azonosítani lehetne és a képernyőn gráfikusan szimulált zongorabillentyűn meg lehetne jeleníteni. Az ötletek a zeneprogramozásban is szabadon szárnyalhatnak. Elképzelhető, hogy nem kell többé zenét programozni? Igen, egészen egyszerűen.

A felhasználónak nem kell többé fáradságos munkával hangjegyeket bevinni, hanem a kompozíciót a billentyűzetről közvetlenül eljátszhatja. Ennek egy három szólamú megoldási módját tartalmazza az 1. függelék.

```
0 REM **** P32. ****
1 :
2 :
3 SYS 49152
4 .OPT OO
5 *=828
10 TONUS = 197    ;VAGY 162
11 OSC1HF  = $D401
12 OSC1PL  = $D402
13 OSC1PH  = $D403
14 OSC1WV  = $D404
15 OSC1AD  = $D405
16 OSC1SR  = $D406
17 HANGERO = $D418
100 SEI:LDA #<IRQ:STA $0314:LDA #>IRQ:STA $0315:CLI:RTS
110 IRQ LDA 197:CMP #64:BEQ GATEOFF
130 LDA #$0F :STA HANGERO
140 LDA 197:AND #$F:STA OSC1PH
150 LDA #$80 :STA OSC1PL
160 LDA TONUS:STA OSC1HF
170 LDA #$41 :STA OSC1WV
```

```
180 LDA #$18 :STA OSC1AD
190 LDA #$58 :STA OSC1SR
200 EXIT JMP $EA31
210 GATEOFF LDA #$40:STA OSC1WV:BNE EXIT
READY.
```

```
0 REM **** P33. ****
1 :
2 :
1000 FOR I=828 TO 893
1010 READ X:POKE I,X:S=S+X:NEXT
1020 DATA 120,169, 73,141, 20,  3,169,  3,141, 21,  3, 88
1030 DATA  96,165,197,201, 64,240, 40,169, 15,141, 24,212
1040 DATA 165,197, 41, 15,141,  3,212,169,128,141,  2,212
1050 DATA 165,197,141,  1,212,169, 65,141,  4,212,169, 24
1060 DATA 141,  5,212,169, 88,141,  6,212, 76, 49,234,169
1070 DATA  64,141,  4,212,208,246
1080 IF S<>7818 THEN PRINT"HIBA A DATA-SORBAN..!!":END
1090 PRINT"OK - INDITAS: SYS 828-AL"
READY.
```

```
0 REM **** P34. ****
1 :
2 :
1000 FOR I=828 TO 893
1010 READ X:POKE I,X:S=S+X:NEXT
1020 DATA 120,169, 73,141, 20,  3,169,  3,141, 21,  3, 88
1030 DATA  96,165,197,201, 64,240, 40,169, 15,141, 24,212
1040 DATA 165,197, 41, 15,141,  3,212,169,128,141,  2,212
1050 DATA 165,162,141,  1,212,169, 65,141,  4,212,169, 24
1060 DATA 141,  5,212,169, 88,141,  6,212, 76, 49,234,169
1070 DATA  64,141,  4,212,208,246
1080 IF S<>7783 THEN PRINT"HIBA A DATA-SORBAN..!!":END
1090 PRINT"OK - INDITAS: SYS 828-AL"
READY.
```

```
0 REM **** P35. ****
1 :
2 :
3 POKE 56,128:CLR:SYS49152
4 .OPT OO
5 * =$8000
10 LSTKEY   = 197
11 OSC1LF   = $D400
12 OSC1HF   = $D401
13 OSC1WV   = $D404
14 OSC1AD   = $D405
15 OSC1SR   = $D406
20 HANGERO = $D418
100 SEI:LDA #<IRQ:STA $0314:LDA #>IRQ:STA $0315:CLI:RTS
110 IRQ LDA LSTKEY:CMP #64:BEQ GATEOFF
120 ASL:TAX
130 LDA TAB,X:STA OSC1LF
140 INX
150 LDA TAB,X:STA OSC1HF
160 LDA #$21:STA OSC1WV
170 LDA #$18:STA OSC1AD
```

```
180 LDA #$58:STA OSC1SR
190 LDA #$0F:STA HANGERO
200 EXIT JMP $EA31
210 GATEOFF LDA #$20:STA OSC1WV:BNE EXIT
999 TAB *=*
1000 DE .WORD $0FD2 ;DELETE
1001 RE .WORD $0218
1002 CR .WORD $3863
1003 F7 .WORD $2187
1004 F1 .WORD $0430
1005 F3 .WORD $0861
1006 F5 .WORD $10C3
1007 CD .WORD $323C
1008 K3 .WORD $04FB
1009 KW .WORD $04B4
1010 KA .WORD $0FD2
1011 K4 .WORD $0000 ;NEM HASZNALT
1012 KZ .WORD $10C3
1013 KS .WORD $11C3
1014 KE .WORD $0547
1015 .WORD $0000
1016 K5 .WORD $05ED
1017 KR .WORD $0598
1018 KD .WORD $13EF
1019 K6 .WORD $06A7
1020 KC .WORD $151F
1021 KF .WORD $0000 ;NEM HASZNALT
1022 KT .WORD $0647
1023 KX .WORD $12D1
1024 K7 .WORD $0777
1025 KY .WORD $070C
1026 KG .WORD $17B5
1027 K8 .WORD $0000 ;NEM HASZNALT
1028 KB .WORD $191E
1029 KH .WORD $1A9C
1030 KU .WORD $07E9
1031 KV .WORD $1660
1032 K9 .WORD $08E1
1033 KI .WORD $0861
1034 KJ .WORD $1DDF
1035 KO .WORD $09F7
1036 KM .WORD $1FA5
1037 KK .WORD $0000 ;NEM HASZNALT
1038 KO .WORD $0968
1039 KN .WORD $1C31
1041 K+ .WORD $0000 ;NEM HASZNALT
1042 KP .WORD $0A8F
1043 KL .WORD $2386
1044 DA .WORD $0BDA ;MINUSZ
1045 PO .WORD $25A2 ;PONT
1046 CO .WORD $270F ;KETTOSPONT
1047 AT .WORD $0B30 ;SZOGLETES ZAROJEL
1048 CM .WORD $2187 ;VESSZO
1049 PF .WORD $0D4E ;FONT-JEL
1050 MU .WORD $0C8F ;CSILLAG
1051 SC .WORD $2CC1 ;PONTOSVESSZO
1052 CL .WORD $0EEF ;CLEAR-HOME
1053 .WORD $0000
```

```
1054 EQ .WORD $2F6B ;EGYENLOSEGJEL
1055 UA .WORD $0E18 ;NYIL FOLFELE
1056 DB .WORD $2A3E ;TORTVONAL
1057 EX .WORD $03F4 ;FELKIALTOJEL
1058 LA .WORD $03BB ;NYIL BALRA
1059 .WORD $0000
1060 QU .WORD $0470 ;HANYAD
1061 .WORD $0000
1062 .WORD $0000
1063 KQ .WORD $0430
1064 RS .WORD $0861 ;RUN-STOP
READY.
```

```
0 REM **** P36. ****
1 :
2 :
1000 POKE 56,128:CLR:FOR I=32768 TO 32959
1010 READ X:POKE I,X:S=S+X:NEXT
1020 DATA 120,169, 13,141, 20,  3,169,128,141, 21,  3, 88
1030 DATA  96,165,197,201, 64,240, 38, 10,170,189, 64,128
1040 DATA 141,  0,212,232,189, 64,128,141,  1,212,169, 33
1050 DATA 141,  4,212,169, 24,141,  5,212,169, 88,141,  6
1060 DATA 212,169, 15,141, 24,212, 76, 49,234,169, 32,141
1070 DATA   4,212,208,246,210, 15, 24,  2, 99, 56,135, 33
1080 DATA  48,  4, 97,  8,195, 16, 60, 50,251,  4,180,  4
1090 DATA 210, 15,  0,  0,195, 16,195, 17, 71,  5,  0,  0
1100 DATA 237,  5,152,  5,239, 19,167,  6, 31, 21,  0,  0
1110 DATA  71,  6,209, 18,119,  7, 12,  7,181, 23,  0,  0
1120 DATA  30, 25,156, 26,233,  7, 96, 22,225,  8, 97,  8
1130 DATA 223, 29,247,  9,165, 31,  0,  0,104,  9, 49, 28
1140 DATA   0,  0,143, 10,134, 35,218, 11,162, 37,223, 39
1150 DATA  48, 11,135, 33, 78, 13,143, 12,193, 44,239, 14
1160 DATA   0,  0,107, 47, 24, 14, 62, 42,244,  3,187,  3
1170 DATA   0,  0,112,  4,  0,  0,  0,  0, 48,  4, 97,  8
1180 IF S<>16114 THEN PRINT"HIBA-A DATA-SORBAN!!":END
1190 PRINT"OK - INDITAS: SYS 32768"
READY.
```

1. FÜGGELÉK:
A Synthimat 64 rövid leírása

Az ötödik fejezet utolsó programjának bevitele után bizonyára észleltük, hogy mennyivel egyszerűbb közvetlenül a billentyűzeten játszani, mint aprólékosan hangjegyről hangjegyre programozni valamilyen zenét.

Az idő múlásával számtalan olyan zeneprogram született, melyek felismerték ezt az előnyt. Azonban egyik sem volt alkalmas a többszólamú játékra. Egyik sem tudott a billentyűzetről akkordokat játszani, vagy basszust és dallamot egyszerre megszólaltatni.

A legtöbb zeneprogram esetében a Commodore 64-es csak egy *szekvenciátor* maradt. A szekvenciátor – hasonlóan a zeneprogramhoz és a zenei segédletekhez – csak arra alkalmas, hogy számára a komponált zenét hangjegyről hangjegyre és időtartamról időtartamra beprogramozzuk. Ez unalmas folyamat, ami gyorsan elkedvetleníti az embert. Meghallgatja egyszer, kétszer, háromszor és ezután az egész elveszti érdekességét.
A Synthimat 64 egy csapásra megszünteti ezeket a hátrányokat. A Commodore 64-est a Synthimat 64 segítségével valódi polifon szintetizátorként lehet használni. Nem kell többé a hangmagasságot és a hangzás időtartamát külön-külön programozni, hanem a billentyűzetről közvetlenül lehet három szólamban játszani.
Kizárólag a funkcióbillentyűk használatával be lehet állítani a hang chip összes paraméterét, valamint egyéb paramétereket is. Ráadásul minden hangot, amit a billentyűzeten lejátszottunk és minden hangszínt, amit bevittünk, lemezen lehet tárolni, amely bármikor újrajátszható vagy törölhető. Így lehetőség nyílik egy saját hangtár létrehozására lemezen anélkül, hogy hangjegyeket, kazettákat kellene készíteni.
A Synthimat 64 csak 8 kbyte hosszúságú és a Profi-Ass 64 segítségével készült. Ezért a program betöltése rövid időt vesz igénybe, mert kevesebb, mint fél perc múlva megjelenik a Commodore 64-es képernyőjén a program kezelőpultja. A képernyőn a hang chip valamennyi eleme logikusan helyezkedik el és más-más színjelöléssel rendelkezik. Minden oszcillátornak megvan a maga színjele. Valamennyi funkció, amely a szóban forgó oszcillátorra vonatkozik, ugyanolyan színjelöléssel van ellátva. A kurzor tetszőlegesen irányítható, mely lehetővé teszi az összes paraméter beállítását.
A Synthimat 64 program teljes mértékben kihasználja a gép billentyűzetét. Két olyan billentyűzet áll rendelkezésre, melyeken egymástól függetlenül lehet játszani. A funkcióbillentyűkkel be lehet állítani, hogy a két billentyűzethez melyik oszcillátort kívánjuk hozzárendelni a háromból. Természetesen valamennyi kombináció szabadon választható.

A képernyőn a program mindkét billentyűzete bármikor láthatóvá tehető. Ezzel mindenkor optikailag is követhető az éppen használatos billentyű. A SHIFT billentyű lenyomásával pedig le lehet kérdezni a billentyűzetet, ha kíváncsiak vagyunk, hogy melyik billentyűhöz melyik hang tartozik.

A Synthimat 64 hangmagasságának beállítása 1/8 félhangok segítségével történhet. Vagyis úgy hangolhatjuk, hogy a hangadás illeszkedjen a saját sztereó berendezésünkhöz vagy más hangszerhez. És mindez nem számok segítségével, hanem a kurzorvezérlő billentyűkkel történik. Oszlopdiagram segítségével a program hangzását optikailag is megjeleníthetjük.

A teljes hangoláson kívül az oszcillátorok egymáshoz viszonyított hangolása is 1/8 félhangonként szabályozható. Ezáltal lebegést is elő tudunk állítani, mellyel szép, széles hangzás érhető el.

Mivel a hang chip korlátozott szólamszámmal (3 szólam) rendelkezik, négy- vagy öthangú akkordokat nem lehet játszani. Ezért a Synthimat 64-ben a két billentyűzet egyikét át lehet kapcsolni akkordautomatikára. Ez lehetővé teszi, hogy egy arpeggio effektus segítségével többszólamú akkordok is lejátszhatók legyenek.
A Synthimat 64 segítségével a lejátszott dalok közvetlenül lemezre is vehetők. A programrész használata: A lemez behelyezése után nyomjuk meg a „felvétel" funkcióját ellátó billentyűt. Ezután folyamatosan vehetjük fel a dalokat. A felvétel befejezéseként pedig a szóköz billentyűt nyomjuk le. Ilyenkor a program a dalt a lemezen tárolja. A felvétel visszahallgatása esetén a „lejátszás" funkcióját ellátó billentyűt nyomjuk meg. Ilyenkor a Commodore 64-es billentyűzetének szerepét a lemezegység veszi át. A dal pontosan úgy ismétlődik meg, ahogy lejátszottuk.

A Synthimat 64 kilenc különböző dal lemezen való tárolásának lehetőségét biztosítja. A program részére a lemezt nem kell külön megformálni, sőt a lemezen más programok is lehetnek.
A program az I/O chip egyidejű (real time) óráját használja, amely szinkronban fut a hálózattal. Az óra a funkcióbillentyűk segítségével bármikor megjeleníthető a képernyő közepén, függetlenül attól, hogy éppen egy dalt játszunk, lemezre készítünk felvételt vagy hangszínt programozunk.

A Synthimat 64 egy ún. belső órát (SYS-órát) is használ, melyet a képernyőn is megjeleníthetünk. Az óra a Synthimat 64 belső sebességét mutatja.
A zeneprogramok legfontosabb kérdése, hogy milyenek a hangzási lehetőségek. A Synthimat 64 az alábbi paraméterekhez fér hozzá.

Paraméter		Érték
VCO 1	hullámforma	8 hullámforma
	fekvés	8 oktáv
	impulzusszélesség	0–4095
LFO 1	hullámforma	8 hullámforma
	sebesség	0–15
LFO 2	hullámforma	8 hullámforma
	sebesség	0–15
Attack – időtartam		0–15
Decay – időtartam		0–15
Sustain – időtartam		0–15
Release – időtartam		0–15

A VCO 2 a VCO 1-nek felel meg LFO 3-mal és 4-gyel.
A VCO 3 a VCO 1-nek felel meg LFO 5-tel és 6-tal.

VCF	beállítások	8 szűrőtípus
	frekvencia	0–2047
	Q érték	0–15
	VCO választás	16 érték
LFO 7	hullámforma	8 hullámforma
	sebesség	0–15
VCA	Hangerő	0–15
LFO 8	hullámforma	8 hullámforma
	sebesség	0–15
SYNC	szinkronizáció	8 módon
RING	gyűrűs moduláció	8 módon

A hang chip valamennyi lehetősége a Synthimat 64 program segítségével kiválasztható és vizuálisan is megjeleníthető.
Az előbbi táblázat 8 LFO-t tartalmaz. Ezek közül az

1,3,5	sorszámú a frekvenciát, a
2,4,6	sorszámú az impulzusszélességet /oszcillátorként egy-egy/, a
7	sorszámú a szűrőfrekvenciát és a
8	sorszámú a hangerőt modulálja.

A billentyűzeten való játék közben, a kurzor mozgatásával valamennyi érték megváltoztatható. A hangzás módosítása azonnal hallhatóvá válik.

Egy tetszés szerinti beállítást bármikor találhatunk hangzás regiszterként a Commodore 64-esen. Egyidejűleg 256(!) regiszter tárolására nyújt lehetőséget a C 64-es memóriája, melyből a kívánt regiszter néhány másodperc alatt kiválasztható. Lehetőség van a regiszter lemezen történő tárolására és visszatöltésére is. A Synthimat 64 segítségével kitárul a szintetikus zene csodavilága!

2. FÜGGELÉK:
A további alkalmazások és a hardver lehetőségek áttekintése

A függelék a C 64-es további zenei alkalmazásait és a különféle hardver lehetőségeket mutatja be.

A C 64-es és a sztereó berendezések összekapcsolása

A C 64-es épp olyan audio jelet állít elő, mint a sztereó kazettás magnetofon. Ezt az audio jelet lehet közvetlenül egy átjátszó kábel segítségével a sztereó berendezéshez vezetni.
De felvetődik a kérdés, hogy hol lehet a hang chip audio jeléhez hozzáférni?
A válasz egyszerű, a jel ki van vezetve a C 64-es hátlapján lévő audio/video aljzat 3. lábára, ami természetesen feleslegessé teszi a hang chip audio out lábának „megcsapolását".
A C 64-es és az átjátszó kábel csatlakozása:
A kábelnek DIN csatlakozóval kell illeszkednie a gép audio/video kimenetéhez. A DIN csatlakozók három- és ötpólusúak. A kábel földvezetékét a középső lábra kell forrasztani, ami az audio/video csatlakozó 2. lába. A DIN csatlakozó lábai félkör alakban helyezkednek el. Ha a gépet hátulról nézzük és megfigyeljük az audio/video aljzatot, akkor azt látjuk, hogy a lábak egy felfelé mutató félkör alakban helyezkednek el. A hang chip audio out jele a baloldali lábon vehető. Ha próbaképpen behelyezzük a DIN csatlakozót az audio/video aljzatba, akkor megláthatjuk, hogy a kábel jelvezetékét a csatlakozó melyik lábára kell kötni.

A sztereó berendezés és az átjátszó kábel csatlakozása: ebben az esetben már bonyolultabb a helyzet, mivel különböző csatlakozó szabványok léteznek. Most a két leggyakoribbal foglalkozunk.

1. A cinch aljzat, mely a legtöbb japán és amerikai berendezésen megtalálható külön csatornánként. Ilyen csatlakozó használata esetén számolni kell azzal, hogy a hang chip hangját csak az egyik hangfalon lehet hallani. A teljes hangzás megteremtésére két lehetőség van:
   a) A készülék sztereo-mono átkapcsolójának mono állásba történő helyezésével. Ezáltal a hang chip hangja mindkét hangfalon hallható lesz.
   b) Két cinch csatlakozó forrasztása az átjátszó kábel másik végére.

2. A DIN csatlakozó, mely a német gyártmányú berendezéseken fordul elő leggyakrabban, csatornánként egy lábbal rendelkezik. A jelet hordozó vezetéket

az ötpólusú DIN csatlakozó megfelelő lábára kell forrasztani, majd a mellette levő két-két lábat össze kell kapcsolni. Így a hangjel mindkét hangfalba eljut.
A sztereó berendezésen hová kell csatlakoztatni a kész kábelt?
A legtöbb készüléken megtalálható az AUX (auxiliary input = segédbemenet.) Ilyen készüléknél a kábelt erre a bemenetre csatlakoztassuk. Egyéb esetben pedig a kazettás magnetofon bemenetét kell igénybe venni.

Külső hangjelek feldolgozása

A hang chip audio in bemenete a Commodore 64-es audio/video csatlakozóján közvetlenül az audio out kivezetés melletti lábra van kötve. Tehát ez az öt láb által képzett félköríven balról a második láb. Ez a láb közvetlenül össze van kötve a hang chip audio in lábával. Így lehetőség van egy külső hangjelnek a hang chipbe táplálására.

Milyen hangjel jöhet itt szóba? Ha pl. erre a bemenetre egy mikrofont csatlakoztatunk, akkor nem fogunk hangot hallani. Ennek az az oka, hogy a jel nincs felerősítve.

Az erősítés legjobban a sztereó berendezéssel végezhető el, mivel az rendelkezik erősítővel. Vegyük példaként külső hangjelnek egy rádióadó hangját. Az erősített jel levételéhez a sztereó berendezésnek azt a kimenetét választjuk, melynek hangerősségét a hangerő szabályozó segítségével be lehet állítani. Ez a kimenet a fejhallgató csatlakozója (a hangszóró melletti kimenet). Számunkra a fejhallgató csatlakozója a legalkalmasabb, mivel a hangszórót nem kell állandóan kihúzni, ha hangjelet akarunk a Commodore 64-esre vezetni.

A feladat tehát a sztereó berendezés fejhallgató csatlakozójának és a Commodore 64-es audio/video csatlakozójának audio in bemenete közötti csatlakoztatás létrehozása.
A fejhallgató csatlakozók a különféle sztereó berendezéseken más és más szabvány szerint készülnek: kis jack csatlakozó (2,5 és 3,5 cm), nagy jack csatlakozó és öt pólusú tuchel csatlakozó. A legegyszerűbb, ha a fejhallgatónkkal elmegyünk egy szaküzletbe, és egy ugyanolyan csatlakozót kérünk, mint amilyen rajta található.

Ha megvan a megfelelő csatlakozó, akkor bontsuk szét a fejhallgató csatlakozóját, és nézzük meg, hogy melyik kivezetésekre van a vezeték ráforrasztva. Ha nem lehet szétbontani a fejhallgató csatlakozóját, ami gyakran előfordulhat, akkor a megfelelő lábat kísérletileg kell megkeresni.
A következő probléma a Commodore 64-es audio/video csatlakozója.
Itt ötpólusú DIN csatlakozóra van szükség. A két csatlakozó összekapcsolásához elegendő egy egyeres árnyékolt vezeték (jel + föld). Forrasszuk a vezeték egyik végét a Commodore 64-es csatlakozójára.

A vezeték másik végénél a helyzet egy kicsit nehezebb, mivel a lábkiosztás készülékenként változó. A sztereó fejhallgatók két jelvezetékkel rendelkeznek, csatornánként eggyel. Ezt a két lábat át kell forrasztani, hogy mono hangjel kerüljön a Commodore 64-esre. A két láb egyikére kell a jelvezetéket forrasztani.

Ha a sztereó berendezés a fejhallgató csatlakoztatásakor automatikusan lekapcsolja a hangszórókat, akkor a vezetéket csak megfelelő hangjel kiválasztása után csatlakoztassuk. Keressünk tehát egy rádióadót, állítsuk alacsony hangerőre, és csatlakoztassuk a vezetéket a sztereó berendezéshez.
Nincs lehetőség arra, hogy a sztereó berendezés hangjelét a Commodore 64-esre vezetésével egy időben a Commodore 64-es jelét visszavezessük a sztereó berendezésre. Ezért az egyik vezetéket el kell távolítani a Commodore 64-es audio/video csatlakozójáról, mielőtt a másikat csatlakoztatnánk. Csak ezután csatlakoztassuk a másik vezetéket. Mivel ilyenkor a video jel nem hozzáférhető, a hang és a kép televízióra vezetésekor hangfrekvencia modulátort használjunk. A Commodore 64-est és a televíziót ne normál vezetékkel, hanem videokábellel kapcsoljuk össze.

Induljunk ki abból, hogy a csatlakoztatás sikeres volt. Ha a Commodore 64-est bekapcsoljuk, a rádióadó nem hallható, még a sztereó erősítő hangerejének maximumán sem. Ennek a jelenségnek nem a sztereó berendezés kis hangteljesítménye az oka. A külső hangjel hangerősségének szabályozása a belső oszcillátorok hangerejéhez hasonlóan a VOL vezérlőbitek (lásd 4. fejezet) segítségével történik. Pl. a maximális hangerő beállításához a POKE 54296,15 utasítást adjuk be. Ezután, ha torzítva is, hallanunk kell a rádióadó hangját a televízió hangszórójában.

Arra is van lehetőség, hogy az audio/video csatlakozóra egy másik kábelt forrasztva a Commodore 64-es audio out jelét egy további sztereó erősítőre, vagy egy kis rádióra vezessük. Sőt a sztereó berendezés kazettás magnetofonján lejátszott kazetta jelét is átadhatjuk a Commodore 64-esnek. A lemezjátszó jele is rávezethető a C 64-esre. Ha van a kazettás magnetofonnak mikrofon bemenete, akkor mikrofon is csatlakoztatható hozzá. Kapcsoljuk a kazettás magnetofont felvétel előkészítés (source vagy monitor) állásba, így beszédhangot is vezethetünk a Commodore 64-esen keresztül a televízió hangszórójára.
Az előbb olvasottak után bizonyára sokakban felvetődik a kérdés: mindez minek? Ehhez nincs szükség a Commodore 64-esre, ezt megtehetjük magával a sztereó berendezéssel is.

Azonban létezik még egy bit, a FILTEX (a 23. regiszter 3. bitje, lásd 4. fejezet), melynek segítségével a külső audio jel átvezethető a szűrőn. És ilyenkor kezdődik el a tényleges játék. A FILTEX bit beállítását végző utasítás: POKE 54295,8. A FILTEX bit törlése a POKE 54295,0 utasítással végezhető el. Ha a bit törölve van, a külső hangjel hangerejét kizárólag a külső hangerővel lehet szabályozni, egyébként változatlan marad. Ha a bit be van állítva, akkor a szűrő 4. fejezetben említett összes lehetősége alkalmazható.